МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ
имени И.Т. ТРУБИЛИНА»

Факультет прикладной информатики
Иностранных языков

УТВЕРЖДЕНО
Декан
Замотайлова Д.А.
27.05.2026

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ) «ИНОСТРАННЫЙ ЯЗЫК. АНГЛИЙСКИЙ ЯЗЫК»

Уровень высшего образования: магистратура

Направление подготовки: 09.04.02 Информационные системы и технологии

Направленность (профиль)подготовки: Проектно-исследовательская деятельность в области информационных технологий

Квалификация (степень) выпускника: магистр

Формы обучения: очная, заочная

Год набора (приема на обучение): 2026

Срок получения образования: Очная форма обучения – 2 года
Заочная форма обучения – 2 года 6 месяца(-ев)

Объем: в зачетных единицах: 7 з.е.
в академических часах: 252 ак.ч.

2026

Разработчики:
Доцент, кафедра иностранных языков Карипиди А.Г.

Рабочая программа дисциплины (модуля) составлена в соответствии с требованиями ФГОС ВО по направлению подготовки 09.04.02 Информационные системы и технологии, утвержденного приказом Минобрнауки от 19.09.2017 № 917, с учетом трудовых функций профессиональных стандартов: "Руководитель разработки программного обеспечения", утвержден приказом Минтруда России от 20.07.2022 № 423н; "Системный администратор информационно-коммуникационных систем", утвержден приказом Минтруда России от 29.09.2020 № 680н; "Специалист по дизайну графических пользовательских интерфейсов", утвержден приказом Минтруда России от 29.09.2020 № 671н; "Специалист по научно-исследовательским и опытно-конструкторским разработкам", утвержден приказом Минтруда России от 04.03.2014 № 121н; "Системный аналитик", утвержден приказом Минтруда России от 27.04.2023 № 367н.

Согласование и утверждение

№	Подразделение или коллегиальный орган	Ответственное лицо	ФИО	Виза	Дата, протокол (при наличии)
1	Иностранных языков	Заведующий кафедрой, руководитель подразделения, реализующего ОП	Непшекуева Т.С.	Согласовано	23.03.2026, № 7

1. Цель и задачи освоения дисциплины (модуля)

Цель освоения дисциплины - Целью освоения дисциплины «Иностранный язык» является обуче-ние практическому владению иностранным языком (английским, немец-ким, французским), критерием которого является умение пользоваться наиболее употребительными языковыми средствами в основных видах ре-чевой деятельности: говорение, аудирование, чтение и письмо.

Задачи изучения дисциплины:

- – научить студентов: представлять результаты научно-исследовательской деятельности в устной и письменной форме на ино-странном языке; ;
- – читать профессиональную литературу на иностранном языке; ана-лизировать, обобщать и критически осмыслять информацию;;
- – обрабатывать большие объемы иноязычной информации; пони-мать оригинальную речь на иностранном языке; логично и связно выска-зываться на профессиональные темы с соблюдением грамматических и фо-нетических норм; ;
- – создавать понятный, грамотный и связный текст, обладающий полнотой изложения и достоверностью, с соблюдением норм, присущих основным жанрам и формам научного дискурса..

2. Планируемые результаты обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы

Компетенции, индикаторы и результаты обучения

УК-4 Способен применять современные коммуникативные технологии, в том числе на иностранном(ых) языке(ах), для академического и профессионального взаимодействия

УК-4.1 Демонстрирует интегративные умения, необходимые для написания, письменного перевода и редактирования различных академических текстов (рефератов, эссе, обзоров, статей т.д.)

Знать:

УК-4.1/Зн1 Интегративные умения, необходимые для написания, письменного перевода и редактирования различных академических текстов (рефератов, эссе, обзоров, статей и т.д.).

Уметь:

УК-4.1/Ум1 Демонстрировать интегративные умения, необходимые для написания, письменного перевода и редактирования различных академических текстов (рефератов, эссе, обзоров, статей и т.д.).

Владеть:

УК-4.1/Нв1 Способностью интегративного умения, необходимого для написания, письменного перевода и редактирования различных академических текстов (рефератов, эссе, обзоров, статей и т.д.).

УК-4.2 Представляет результаты академической и профессиональной деятельности на различных научных мероприятиях, включая международные

Знать:

УК-4.2/Зн1 Результаты академической и профессиональной деятельности на различных научных мероприятиях, включая международные.

Уметь:

УК-4.2/Ум1 Представлять результаты академической и профессиональной деятельности на различных научных мероприятиях, включая международные.

Владеть:

УК-4.2/Нв1 Способностью представлять результаты академической и профессиональной деятельности на различных научных мероприятиях, включая международные.

УК-4.3 Демонстрирует интегративные умения, необходимые для эффективного участия в академических и профессиональных дискуссиях

Знать:

УК-4.3/Зн1 Интегративные умения, необходимые для эффективного участия в академических и профессиональных дискуссиях.

Уметь:

УК-4.3/Ум1 Демонстрировать интегративные умения, необходимые для эффективного участия в академических и профессиональных дискуссиях.

Владеть:

УК-4.3/Нв1 Способностью интегративного умения, необходимого для эффективного участия в академических и профессиональных дискуссиях.

3. Место дисциплины в структуре ОП

Дисциплина (модуль) «Иностранный язык» относится к обязательной части образовательной программы и изучается в семестре(ах): Очная форма обучения - 1, 2, Заочная форма обучения - 1, 2.

В процессе изучения дисциплины студент готовится к решению типов задач профессиональной деятельности, предусмотренных ФГОС ВО и образовательной программой.

4. Объем дисциплины (модуля) и виды учебной работы

Очная форма обучения

Период обучения	Общая трудоемкость (часы)	Общая трудоемкость (ЗЕТ)	Контактная работа (часы, всего)	Внеаудиторная контактная работа (часы)	Зачет (часы)	Практические занятия (часы)	Самостоятельная работа (часы)	Промежуточная аттестация (часы)
Первый семестр	108	3	31	1		30	77	Зачет
Второй семестр	144	4	33	3		30	84	Экзамен (27)
Всего	252	7	64	4		60	161	27

Заочная форма обучения

Период	доемкость сы)	доемкость ЗТ)	ая работа всего)	я контактная (часы)	(часы)	ие занятия сы)	ие занятия сы)	ьная работа сы)	ая аттестация сы)

обучения	Общая тру (ча	Общая тру (ЗЕ	Контактн (часы,	Внеаудиторна работа	Зачет	Лекционны (ча	Практическ (ча	Самостоятел (ча	Промежуточн (ча
Первый семестр	108	3	15	1	4	4	6	93	Зачет (4)
Второй семестр	144	4	17	3			14	118	Экзамен (9)
Всего	252	7	32	4	4	4	20	211	9

5. Содержание дисциплины (модуля)

5.1. Разделы, темы дисциплины и виды занятий

(часы промежуточной аттестации не указываются)

Очная форма обучения

Наименование раздела, темы	Всего	Внеаудиторная контактная работа	Практические занятия	Самостоятельная работа	Планируемые результаты обучения, соотнесенные с результатами освоения программы
Раздел 1. Компьютерная безопастность в английском языке	**107**		**30**	**77**	УК-4.1 УК-4.2 УК-4.3
Тема 1.1. Computer Security	53		15	38	
Тема 1.2. Cloud Computing	54		15	39	
Раздел 2. Промежуточная аттестация	**1**	**1**			УК-4.1 УК-4.2 УК-4.3
Тема 2.1. Зачет	1	1			
Раздел 3. Искусственный интеллект в английском языке	**114**		**30**	**84**	УК-4.1 УК-4.2 УК-4.3
Тема 3.1. Artifical intelligence	28		6	22	
Тема 3.2. Preshing on programming	28		6	22	
Тема 3.3. Recruitment	26		6	20	
Тема 3.4. Business Documents	32		12	20	
Раздел 4. Промежуточная аттестация	**3**	**3**			УК-4.1 УК-4.2 УК-4.3
Тема 4.1. Экзамен	3	3			
Итого	**225**	**4**	**60**	**161**	

Заочная форма обучения

		гактная	я	ции	абота	ьтаты нные с ния

Наименование раздела, темы	Всего	Внеаудиторная конт работа	Лекционные занятия	Практические занят	Самостоятельная ра	Планируемые резул обучения, соотнесе результатами освое программы
Раздел 1. Компьютерная безопастность в английском языке	**50**		**4**	**6**	**40**	УК-4.1 УК-4.2 УК-4.3
Тема 1.1. Computer Security	26		2	4	20	
Тема 1.2. Cloud Computing	24		2	2	20	
Раздел 2. Промежуточная аттестация	**1**	**1**				УК-4.1 УК-4.2 УК-4.3
Тема 2.1. Зачет	1	1				
Раздел 3. Искусственный интеллект в английском языке	**185**			**14**	**171**	УК-4.1 УК-4.2 УК-4.3
Тема 3.1. Artifical intelligence	54			4	50	
Тема 3.2. Preshing on programming	54			4	50	
Тема 3.3. Recruitment	35			2	33	
Тема 3.4. Business Documents	42			4	38	
Раздел 4. Промежуточная аттестация	**3**	**3**				УК-4.1 УК-4.2 УК-4.3
Тема 4.1. Экзамен	3	3				
Итого	**239**	**4**	**4**	**20**	**211**	

5.2. Содержание разделов, тем дисциплин

Раздел 1. Компьютерная безопастность в английском языке
(Заочная: Лекционные занятия - 4ч.; Практические занятия - 6ч.; Самостоятельная работа - 40ч.; Очная: Практические занятия - 30ч.; Самостоятельная работа - 77ч.)

Тема 1.1. Computer Security
(Заочная: Лекционные занятия - 2ч.; Практические занятия - 4ч.; Самостоятельная работа - 20ч.; Очная: Практические занятия - 15ч.; Самостоятельная работа - 38ч.)
Verbals (theory)

Тема 1.2. Cloud Computing
(Заочная: Лекционные занятия - 2ч.; Практические занятия - 2ч.; Самостоятельная работа - 20ч.; Очная: Практические занятия - 15ч.; Самостоятельная работа - 39ч.)
Gerund

Раздел 2. Промежуточная аттестация
(Заочная: Внеаудиторная контактная работа - 1ч.; Очная: Внеаудиторная контактная работа - 1ч.)

Тема 2.1. Зачет
(Заочная: Внеаудиторная контактная работа - 1ч.; Очная: Внеаудиторная контактная работа - 1ч.)
По итогам изученного курса обучающиеся сдают зачет

Раздел 3. Искусственный интеллект в английском языке

(Заочная: Практические занятия - 14ч.; Самостоятельная работа - 171ч.; Очная: Практические занятия - 30ч.; Самостоятельная работа - 84ч.)

Тема 3.1. Artifical intelligence
(Заочная: Практические занятия - 4ч.; Самостоятельная работа - 50ч.; Очная: Практические занятия - 6ч.; Самостоятельная работа - 22ч.)
Participle

Тема 3.2. Preshing on programming
(Заочная: Практические занятия - 4ч.; Самостоятельная работа - 50ч.; Очная: Практические занятия - 6ч.; Самостоятельная работа - 22ч.)
The Infinitive

Тема 3.3. Recruitment
(Заочная: Практические занятия - 2ч.; Самостоятельная работа - 33ч.; Очная: Практические занятия - 6ч.; Самостоятельная работа - 20ч.)
Набор персонала

Тема 3.4. Business Documents
(Заочная: Практические занятия - 4ч.; Самостоятельная работа - 38ч.; Очная: Практические занятия - 12ч.; Самостоятельная работа - 20ч.)
Деловые документы

Раздел 4. Промежуточная аттестация
(Заочная: Внеаудиторная контактная работа - 3ч.; Очная: Внеаудиторная контактная работа - 3ч.)

Тема 4.1. Экзамен
(Заочная: Внеаудиторная контактная работа - 3ч.; Очная: Внеаудиторная контактная работа - 3ч.)
Итоговая проверка знаний

6. Оценочные материалы текущего контроля

Раздел 1. Компьютерная безопастность в английском языке

Форма контроля/оценочное средство: Задача

Вопросы/Задания:

1. What do the principles of effective writing include?

1. brevity
2. clarity
3. accuracy
4. brevity, clarity and accuracy

2. Which is the best program to use to write a letter?

1. Microsoft Word
2. Microsoft Excel
3. Microsoft Note
4. Adobe Photoshop

3. Match English and Russian equivalents:

Application program - Программное обеспечение
Developer - Равный

Equal - Разработчик
Software - Прикладная программа
Apply - Использовать

4. Identify the best type/types of supporting evidence that you can use in a presentation:

1. Statistical Data
2. Quotes from People or Books
3. Client Testimonials
4. Wikipedia Sources

5. Who created Microsoft?

Who created Microsoft?

6. Put the words in the right order:

Crucial - 1
Security - 2
confidential - 3
is - 4
online - 5
when - 6
information - 7
send - 8
you - 9

7. It … sufficient to recognize that projects and conflict are inseparable companions.

It … sufficient to recognize that projects and conflict are inseparable companions.

8. What type of communication is a report to a customer from an employer?

1. manager communication
2. administrator communication
3. official communication
4. officer communication

9. Decide the best response to your interviewer’s question: Do you like working with your current boss?

1. No. I think he can’t manage people.
2. No. He is too aggressive and lazy.
3. No. However, I’ve learnt a lot from him.

10. The most common methods of protection are:

1. passwords for access control
2. firewalls
3. encryption
4. software
5. decryption systems

11. Match the synonyms:

Rivalry - management
administration - competition
convention - conference
diversity - value
worth - variety

12. Put the words in the right order:

To fraud, - 1
passwords - 2
of privacy - 3
Weak - 4
theft - 5
open - 6
identity - 7
the door - 8

and breaches - 9

13. Choose the right pace of speech at the presentation:

1. about 20% more slowly than normal
2. just as fast as in an normal conversation
3. faster than in a normal conversation

14. Choose the right way to express enthusiasm?

1. by raising voice level
2. by waving arms
3. by making hand or arm gestures for important points

15. Match the words with the definitions:

Query - a search that locates all information of a specific type a single page of a spreadsheet
database - a large group of data organized in a computer system
worksheet - a search that locates all information of a specific type
table - a grid that organizes data into columns and rows
function - a mathematical instruction that performs a specific calculation in a spreadsheet

16. After a meeting with a contact, in order to express your thanks, it is appropriate to:

1. Send him/her a small box of chocolates with a note
2. Drop by the office and give him/her a hot cup of coffee
3. Send a dozen red roses to his/her home
4. Send a thank you letter

17. … Andy having dinner with his friend or his colleague?

1. did
2. do
3. does
4. was

18. ... you be playing the violin or the guitar?

1. was
2. are
3. will
4. do

19. … the flowers been watered today?

1. have
2. had
3. did
4. do

20. … many accidents been caused by careless driving?

1. has
2. have
3. had
4. will have

21. The recent boom in consumer spending resulted in sales … .

1. recession
2. growth
3. improvement
4. decline

22. Fixed assets are what a company … .

1. owes
2. owns
3. sells
4. buys

23. The goods which a company intends to sell to its customers areknown as a … .

1. stock

2. capital
3. stake
4. proficiency

24. We need to … four new people for our office in Manchester.

1. join
2. recruit
3. fire
4. sack

25. We have 200 people on our … .

1. recruitment
2. business
3. payroll
4. band

26. The CEO is the head of the … team.

1. government
2. management
3. venture
4. legislation

27. You can buy the same software quite … at our local market.

1. reasonably
2. cheaply
3. freely
4. costly

28. New technologies make global … easier.

1. communication
2. travelling
3. lack
4. resistance

29. The mother told the children that she … them to come for lunch just in time.

1. hadn't expected
2. doesn't expect
3. didn't expect
4. wouldn’t

30. At the moment your candidacy … by commission of experts.

1. has not been considered
2. is not being considered
3. was not considered
4. has not considered

31. Mary ... for her examination for the whole August.

1. won't be preparing
2. won't prepare
3. prepare
4. prepares

32. The manager... an announcement to the press later this afternoon.

1. won’t be making
2. won’t be made
3. doesn’t make
4. didn’t make

Раздел 2. Промежуточная аттестация

Форма контроля/оценочное средство:

Вопросы/Задания:

.

Раздел 3. Искусственный интеллект в английском языке

Форма контроля/оценочное средство: Задача

Вопросы/Задания:

1. What is the other name for Spreadsheet?

1. Excel
2. Word
3. PowerPoint
4. Photoshop

2. Match Russian and English equivalents:

USB-накопитель - Scanner
Снимок экрана - AdBlock
Сканер - USB flash drive
Гибкая разработка программ - Agile Software Development
Блокировка рекламы - Screenshot

3. How should you start an email?

1. Write your name
2. Put a stamp on it
3. Put the date
4. Write the subject

4. Match English and Russian equivalents:

Browser - Интернет
Internet - Браузер
Label - Хранилище
Storage - Ярлык
Archive - Архив

5. Advanced telecommunications and computer technology allow such virtual projects to … created.

Advanced telecommunications and computer technology allow such virtual projects to … created.

6. How much time should you spend on each slide?

1. Any length of time
2. 30 seconds
3. 2-3 minutes
4. 5 minutes

7. Match English and Russian equivalents:

Random access memory - оперативная память
Modem - интерфейс
Interface - модем
Display - воспроизведение
Join - спам
Spam - соединение
Log in - вход

8. Put the words in the right order:

Information - 1
without - 2
Spyware - 3
consent - 4
from - 5
collects - 6
PC - 7
your - 8
your - 9

9. Choose the best way to deal with nervousness (What should you do when you feel nervous?):

1. hold a pen or cards in your hands
2. walk back and forth
3. look at the flip chart or screen (not at the audience)

10. Identify the true statement: «Public Administration may be defined as …»:

1. Administrative Capacity
2. Management of industry
3. Management of Property
4. Administration of Public

11. What does internet security do to help your computer?

1. It clears all the viruses and trojans from your system.
2. It keeps you from downloading internet files.
3. It stops you from accessing the internet.
4. It messes with your computers operating system.

12. Choose the features a summary should have:

1. The author’s name and the title of the article
2. The author’s main idea
3. Details to support the idea
4. Quotations

13. Put the words in the right order:

Attachments - 1
through - 2
program - 3
A worm - 4
a selfcopying - 5
spreads - 6
is - 7
email - 8
that - 9

14. It is generally NOT a good practice to deliver a presentation by:

1. Reading the entire presentation line by line
2. Elaborating each bullet point on the presentation materials
3. Paraphrasing what is on the presentation materials
4. Mentioning the highlights of what is on the screen

15. Match the words with their definitions.

Font - the style of the typed characters on a document
compatibility - the ability of one program or file to work with a different program
peripheral - gathering raw data
webcam - a device that broadcasts sound and video on the Net
data collection - a device that can be connected to a computer

16. Which of the factors below should you consider while publishing a scientific paper?

1. The scope and aims of the journal are relevant to your paper.
2. The journal impact factor is reasonably high.
3. The page charges are low.
4. The time take to accept or reject a paper is quite short.

17. That's amazing! She … fifteen kilometers this morning!

1. have run
2. has run
3. had run
4. had been run

18. If we had practiced more, we … the competition.

1. would win
2. would have won
3. will win
4. won

19. My spirit though ...was not broken.

1. crushed
2. crushing
3. being crushed
4. was crushed

20. I like to study foreign languages. German is ... .

1. fascinated
2. fascinating
3. being fascinated
4. fascinate

21. Your dress wants … .

1. clean
2. to clean
3. being cleaned
4. cleaning

22. I am going to … my trip to New-York. The situation changed completely.

1. deny
2. refuse
3. cancel
4. reject

23. We decided to … the new model at the trade fair in autumn.

1. manufacture
2. produce
3. launch
4. build

24. He … most of his savings in the StockExchange.

1. gave
2. fixed
3. invested
4. anticipated

25. He is proud of never … at chess by his fellow students.

1. beating
2. having beaten
3. being beaten
4. having been beaten

26. I remember … her the letter.

1. of showing
2. being shown
3. having shown
4. having been shown

27. The passengers looked at the bus in surprise as though ... that it had happened.

1. not having believed
2. not believing
3. not believed
4. don`t believe

28. I saw Sofia. Why … for you at the library?

1. was she waiting
2. shewas waiting

3. has she waited
4. has she been waited

29. Don’t be noisy, the children… .

1. are sleeping
2. were sleeping
3. was sleeping
4. sleeping

30. Did you say … very early the following morning?

1. will you have to pass your exam
2. you would have to pass your exam
3. would you have to pass your exam
4. you will have to pass your exam

31. He said he … always remember me.

1. would always remember
2. will always remember
3. had always remembered
4. will be always remember

32. If the athlete … his speed, he won’t break the record.

1. hadn’t improved
2. doesn’t improve
3. won’t improve
4. didn’t improve

Раздел 4. Промежуточная аттестация

Форма контроля/оценочное средство:

Вопросы/Задания:

.

7. Оценочные материалы промежуточной аттестации

Очная форма обучения, Первый семестр, Зачет
Контролируемые ИДК: УК-4.1 УК-4.2 УК-4.3
Вопросы/Задания:

1. How worried are you about Internet security?

2. Have you ever had computer problems?

3. What risks are there of being online?

4. What do you think about pop-up windows?

5. What is malware?

6. What do you think about this scam?

7. Could fraudsters one day make the Internet too dangerous?

8. What do you think of the idea of passwords?

9. What do you think of the Web Authentication system?

10. What do you know about biometrics?

11. How secure do you think fingerprints and facial recognition are?

12. Have you ever been hacked?

13. How dangerous is the Internet?

14. What happens when someone steals your identity?

15. What is your biggest worry about being online?

16. What advice do you have for people to stay safe online?

17. What do you know about facial recognition?

18. What are the advantages and disadvantages of facial recognition?

19. Should facial recognition be everywhere?

20. Do you think police officers should wear body cameras?

21. How else can police find criminals quickly?

22. What happens to cities with outdated technology?

23. Would you like to see facial recognition in your town?

24. What do you know about Big Brother?

25. Why are people afraid of the idea of Big Brother?

26. How do we go about making our online transactions secure?

27. What is cloud computing?

28. What is programming?

29. What programming languages do you know?

30. What is flash memory?

31. What is a computer?

32. What is data?

33. What are main functions of a computer?

34. Give examples of using computers in everyday life.

35. What types of computers do you know?

36. What is the role of computers in our sour society?

37. What is a program?

38. What is a computer system?

39. What is a hardware?

40. What is a User Interface?

41. What is information technology?

42. Give the definition of software.

43. Are you a research student?

44. What department are you in?

45. Who Is your scientific adviser (supervisor) of your master's work?

46. What is the subject of your research?

47. What problems does your master's work concentrate on?

48. What problems does your current research include?

49. What methods do you apply in your research? Do you use any new technologies?

50. What web sites do you use for research work?

51. Have you any publications on the subject you study?

52. Where do you work now and as what? Is your work related to computer?

53. What are the research interests of your supervisor?

54. What is the difference between the first computer and the modern?

55. How do computers treat people?

56. How do computers help people?

57. What is the Internet?

58. How can the Internet be useful educational purposes?

59. Why do people want to be “on the internet”?

60. What does a computer virus cause?

Очная форма обучения, Второй семестр, Экзамен
Контролируемые ИДК: УК-4.1 УК-4.2 УК-4.3
Вопросы/Задания:

1. How worried are you about Internet security?

2. Have you ever had computer problems?

3. What risks are there of being online?

4. What do you think about pop-up windows?

5. What is malware?

6. What do you think about this scam?

7. Could fraudsters one day make the Internet too dangerous?

8. What do you think of the idea of passwords?

9. What do you think of the Web Authentication system?

10. What do you know about biometrics?

11. How secure do you think fingerprints and facial recognition are?

12. Have you ever been hacked?

13. How dangerous is the Internet?

14. What happens when someone steals your identity?

15. What is your biggest worry about being online?

16. What advice do you have for people to stay safe online?

17. What do you know about facial recognition?

18. What are the advantages and disadvantages of facial recognition?

19. Should facial recognition be everywhere?

20. Do you think police officers should wear body cameras?

21. How else can police find criminals quickly?

22. What happens to cities with outdated technology?

23. Would you like to see facial recognition in your town?

24. What do you know about Big Brother?

25. Why are people afraid of the idea of Big Brother?

26. How do computers treat people?

27. How do computers help people?

28. What is the Internet?

29. How can the Internet be useful educational purposes?

30. Why do people want to be “on the internet”?

31. What does a computer virus cause?

32. What dangerous are there for a society which depends on computer screens rather than face-to-face contact for its main means of communications

33. What direction in the magistracy did you choose? Why?

34. Why did you decide to enter the magistracy and in what direction?

35. What should you study more computer science (informatics) or math at your faculty?

36. What does pervasive computing mean?

37. What do you think – How has the computer changed the world?

38. What risks do hackers pose for Internet users?

39. How can one avoid risks using Internet?

40. What way is there to protect a message?

41. What are the most common methods of private networks protection?

42. What are Malwares (Malicious Software)?

43. What is a worm?

44. What is the origin of the word ‘hacker’?

45. Who are ‘white hats’ in computer industry?

46. What password can be considered secure?

47. The Internet guides people to the one, open society, doesn't it? Give examples. What cloud computing services are available?

48. Give examples of cloud computing.

49. What is the history of cloud computing?

50. Who is the hyper scale cloud computing provider?

51. What do abbreviations AT and ML stand for?

52. Why are Al and ML experiencing a strong surge of interest recently'?

53. What language can computers understand?

54. What languages are called 'low-level' languages?

55. What languages are called 'high-level' languages?

56. Can you name some 'high-level' languages?

57. What is a program?

58. What is information technology?

59. What department are you in?

60. What web sites do you use for research work?

61. Text 1

Internet of Things

What if those new jeans you've just bought start tweeting about your location as you cross London Bridge? It sounds bizarre, but it's possible – if they are equipped with a tiny RFID device, your location could be revealed without you knowing about it. This technology is just one of the current ways of allowing physical objects to go online – the so-called Internet of Things. Those in favour of the IoT claim that interconnectivity would allow us to locate and monitor everything, everywhere and at any time. Imagine a smart building where you know how many people are inside just by detecting movement with motion-sensitive lights. This could help save lives in an emergency. But as more objects become part of the digital world, there is growing debate over the benefits of smart technology versus the lack of privacy. To what extent can surveillance of people be accepted? Which principles should govern the use of the IoT? The European Commission, for example, has established a framework to safeguard consumer privacy as industries develop this technology further. Within the retail industry, a number of stores have started using RFID tags to check and track stock more easily. However, some people are worried that the RFID reader being used by a shop employee to check the number of pairs of jeans could also read the data on a customer's driving license, for example, if it contained a RFID chip. This could then lead to identity theft. If the tag is not removed at the checkout, the item could be tracked on the street. Once the tag is thrown away, it can still be scanned, allowing someone to get an idea of your shopping habits. Supporters of the IoT point out

that in our already digital and high-tech society your mobile phone operator and bank know much more about your life than your partner does and it is certainly more critical information than the type of jeans you wear.

62. Text 2

Information

Information can be thought of as the resolution of uncertainty; it is that which answers the question of "what an entity is" and thus defines both its essence and nature of its characteristics. It is associated with data, as data represents values attributed to parameters, and information is data in context and with meaning attached. Information relates also to knowledge, as knowledge signifies understanding of an abstract or concrete concept.

In terms of communication, information is expressed either as the content of a message or through direct or indirect observation. That which is perceived can be construed as a message in its own right, and in that sense, information is always conveyed as the content of a message.

Information can be encoded into various forms for transmission and interpretation (for example, information may be encoded into a sequence of signs, or transmitted via a signal). It can also be encrypted for safe storage and communication.

The uncertainty of an event is measured by its probability of occurrence and is inversely proportional to that. The more uncertain an event, the more information is required to resolve uncertainty of that event. The bit is a typical unit of information, but other units such as the natural unit of information may be used. For example, the information encoded in one "fair" coin flip is log2(2/1) = 1 bit, and in two fair coin flips is log2(4/1) = 2 bits.

The concept of information has different meanings in different contexts. Thus the concept becomes related to notions of constraint, communication, control, data, form, education, knowledge, meaning, understanding, mental stimuli, pattern, representation, and entropy.

63. Text 3

C++ library

The C++ standard consists of two parts: the core language and the standard library. C++ programmers expect the latter on every major implementation of C++; it includes aggregate types (vectors, lists, maps, sets, queues, stacks, arrays, tuples), algorithms (find, for each, binary search, random shuffle, etc.), input/output facilities (iostream, for reading from and writing to the console and files), filesystem library, localization support, smart pointers for automatic memory management, regular expression support, multi-threading library, atomics support (allowing a variable to be read or written to by at most one thread at a time without any external synchronization), time utilities (measurement, getting current time, etc.), a system for converting error reporting that doesn't use C++ exceptions into C++ exceptions, a random number generator and a slightly modified version of the C standard library (to make it comply with the C++ type system).

A large part of the C++ library is based on the Standard Template Library (STL). Useful tools provided by the STL include containers as the collections of objects (such as vectors and lists), iterators that provide array-like access to containers, and algorithms that perform operations such as searching and sorting.

Furthermore, (multi)maps (associative arrays) and (multi)sets are provided, all of which export compatible interfaces. Therefore, using templates it is possible to write generic algorithms that work with any container or on any sequence defined by iterators. As in C, the features of the library are accessed by using the #include directive to include a standard header. The C++ Standard Library provides 105 standard headers, of which 27 are deprecated.

64. Text 4

Static polymorphism

Function overloading allows programs to declare multiple functions having the same name but with different arguments. The functions are distinguished by the number or types of their formal parameters. Thus, the same function name can refer to different functions depending on the context in which it is used. The type returned by the function is not used to distinguish overloaded functions and would result in a compile-time error message.

When declaring a function, a programmer can specify for one or more parameters a default value. Doing so allows the parameters with defaults to optionally be omitted when the function is called, in

which case the default arguments will be used. When a function is called with fewer arguments than there are declared parameters, explicit arguments are matched to parameters in left-to-right order, with any unmatched parameters at the end of the parameter list being assigned their default arguments. In many cases, specifying default arguments in a single function declaration is preferable to providing overloaded function definitions with different numbers of parameters.
Templates in C++ provide a sophisticated mechanism for writing generic, polymorphic code (i.e. parametric polymorphism). In particular, through the curiously recurring template pattern, it's possible to implement a form of static polymorphism that closely mimics the syntax for overriding virtual functions. Because C++ templates are type-aware and Turing-complete, they can also be used to let the compiler resolve recursive conditionals and generate substantial programs through template metaprogramming. Contrary to some opinion, template code will not generate a bulk code after compilation with the proper compiler settings.

65. Text 5

CPU

A central processing unit (CPU), also called a central processor or main processor, is the electronic circuitry within a computer that executes instructions that make up a computer program. The CPU performs basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions. The computer industry has used the term "central processing unit" at least since the early 1960s. Traditionally, the term "CPU" refers to a processor, more specifically to its processing unit and control unit (CU), distinguishing these core elements of a computer from external components such as main memory and I/O circuitry.
The form, design, and implementation of CPUs have changed over the course of their history, but their fundamental operation remains almost unchanged. Principal components of a CPU include the arithmetic logic unit (ALU) that performs arithmetic and logic operations, processor registers that supply operands to the ALU and store the results of ALU operations, and a control unit that orchestrates the fetching (from memory) and execution of instructions by directing the coordinated operations of the ALU, registers and other components.
Most modern CPUs are microprocessors, where the CPU is contained on a single metal-oxide-semiconductor (MOS) integrated circuit (IC) chip. An IC that contains a CPU may also contain memory, peripheral interfaces, and other components of a computer; such integrated devices are variously called microcontrollers or systems on a chip (SoC). Some computers employ a multi-core processor, which is a single chip containing two or more CPUs called "cores"; in that context, one can speak of such single chips as "sockets".

66. Text 6

Data organization

A modern digital computer represents data using the binary numeral system. Text, numbers, pictures, audio, and nearly any other form of information can be converted into a string of bits, or binary digits, each of which has a value of 1 or 0. The most common unit of storage is the byte, equal to 8 bits. A piece of information can be handled by any computer or device whose storage space is large enough to accommodate the binary representation of the piece of information, or simply data. For example, the complete works of Shakespeare, about 1250 pages in print, can be stored in about five megabytes (40 million bits) with one byte per character.
Data are encoded by assigning a bit pattern to each character, digit, or multimedia object. Many standards exist for encoding
By adding bits to each encoded unit, redundancy allows the computer to both detect errors in coded data and correct them based on mathematical algorithms. Errors generally occur in low probabilities due to random bit value flipping, or "physical bit fatigue", loss of the physical bit in storage of its ability to maintain a distinguishable value (0 or 1), or due to errors in inter or intra-computer communication. A random bit flip (due to random radiation) is typically corrected upon detection. A bit, or a group of malfunctioning physical bits (not always the specific defective bit is known; group definition depends on specific storage device) is typically automatically fenced-out, taken out of use by the device, and replaced with another functioning equivalent group in the device, where the corrected bit values are restored (if possible). The cyclic redundancy check (CRC) method is

typically used in communications and storage for error detection. A detected error is then retried.

67. Text 7

Linux-V server

Linux-VServer is a virtual private server implementation that was created by adding operating system-level virtualization capabilities to the Linux kernel. It is developed and distributed as open-source software.

The project was started by Jacques Gélinas. It is now maintained by Herbert Pötzl. It is not related to the Linux Virtual Server project, which implements network load balancing.

Linux-VServer is a jail mechanism in that it can be used to securely partition resources on a computer system (such as the file system, CPU time, network addresses and memory) in such a way that processes cannot mount a denial-of-service attack on anything outside their partition.

Each partition is called a security context, and the virtualized system within it is the virtual private server. A chroot-like utility for descending into security contexts is provided. Booting a virtual private server is then simply a matter of kickstarting init in a new security context; likewise, shutting it down simply entails killing all processes with that security context. The contexts themselves are robust enough to boot many Linux distributions unmodified, including Debian and Fedora.

Virtual private servers are commonly used in web hosting services, where they are useful for segregating customer accounts, pooling resources and containing any potential security breaches. To save space on such installations, each virtual server's file system can be created as a tree of copy-on-write hard links to a "template" file system. The hard link is marked with a special filesystem attribute and when modified, is securely and transparently replaced with a real copy of the file.

68. Text 8

Computer engineering

Computer engineering (CE) is a branch of engineering that integrates several fields of computer science and electronic engineering required to develop computer hardware and software. Computer engineers usually have training in electronic engineering (or electrical engineering), software design, and hardware-software integration instead of only software engineering or electronic engineering. Computer engineers are involved in many hardware and software aspects of computing, from the design of individual microcontrollers, microprocessors, personal computers, and supercomputers, to circuit design. This field of engineering not only focuses on how computer systems themselves work but also how they integrate into the larger picture.

Usual tasks involving computer engineers include writing software and firmware for embedded microcontrollers, designing VLSI chips, designing analog sensors, designing mixed signal circuit boards, and designing operating systems. Computer engineers are also suited for robotics research, which relies heavily on using digital systems to control and monitor electrical systems like motors, communications, and sensors.

In many institutions of higher learning, computer engineering students are allowed to choose areas of in-depth study in their junior and senior year because the full breadth of knowledge used in the design and application of computers is beyond the scope of an undergraduate degree. Other institutions may require engineering students to complete one or two years of general engineering before declaring computer engineering as their primary focus.

69. Text 9

Computer programming

Computer programming is the process of designing and building an executable computer program for accomplishing a specific computing task. Programming involves tasks such as: analysis, generating algorithms, profiling algorithms' accuracy and resource consumption, and the implementation of algorithms in a chosen programming language (commonly referred to as coding). The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit. The purpose of programming is to find a sequence of instructions that will automate the performance of a task (which can be as complex as an operating system) on a computer, often for solving a given problem. The process of programming thus often requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.

Tasks accompanying and related to programming include: testing, debugging, source code maintenance, implementation of build systems, and management of derived artifacts, such as the machine code of computer programs. These might be considered part of the programming process, but often the term software development is used for this larger process with the term programming, implementation, or coding reserved for the actual writing of code. Software engineering combines engineering techniques with software development practices. Reverse engineering is the opposite process. A hacker is any skilled computer expert that uses their technical knowledge to overcome a problem, but it can also mean a security hacker in common language.

70. Text 10

Machine code

Machine code is a computer program written in machine language instructions that can be executed directly by a computer's central processing unit (CPU). Each instruction causes the CPU to perform a very specific task, such as a load, a store, a jump, or an ALU operation on one or more units of data in CPU registers or memory.

Machine code is a strictly numerical language which is intended to run as fast as possible, and may be regarded as the lowest-level representation of a compiled or assembled computer program or as a primitive and hardware-dependent programming language. While it is possible to write programs directly in machine code, it is tedious and error prone to manage individual bits and calculate numerical addresses and constants manually. For this reason, programs are very rarely written directly in machine code in modern contexts, but may be done for low level debugging, program patching (especially when assembler source is not available) and assembly language disassembly.

The overwhelming majority of practical programs today are written in higher-level languages or assembly language. The source code is then translated to executable machine code by utilities such as compilers, assemblers, and linkers, with the important exception of interpreted programs, which are not translated into machine code. However, the interpreter itself, which may be seen as an executor or processor, performing the instructions of the source code, typically consists of directly executable machine code (generated from assembly or high-level language source code).

Machine code is by definition the lowest level of programming detail visible to the programmer, but internally many processors use microcode or optimize and transform machine code instructions into sequences of micro-ops. This is not generally considered to be a machine code.

71. Text 11

Computer software

Computer software, or simply software, is a collection of data or computer instructions that tell the computer how to work. This is in contrast to physical hardware, from which the system is built and actually performs the work. In computer science and software engineering, computer software is all information processed by computer systems, programs and data. Computer software includes computer programs, libraries and related non-executable data, such as online documentation or digital media. Computer hardware and software require each other and neither can be realistically used on its own.

At the lowest programming level, executable code consists of machine language instructions supported by an individual processor—typically a central processing unit (CPU) or a graphics processing unit (GPU). A machine language consists of groups of binary values signifying processor instructions that change the state of the computer from its preceding state. For example, an instruction may change the value stored in a particular storage location in the computer—an effect that is not directly observable to the user. An instruction may also invoke one of many input or output operations, for example displaying some text on a computer screen; causing state changes which should be visible to the user. The processor executes the instructions in the order they are provided, unless it is instructed to "jump" to a different instruction, or is interrupted by the operating system. As of 2015, most personal computers, smartphone devices and servers have processors with multiple execution units or multiple processors performing computation together, and computing has become a much more concurrent activity than in the past.

72. Text 12

Input/output

In computing, input/output or I/O (or, informally, io or IO) is the communication between an information processing system, such as a computer, and the outside world, possibly a human or another information processing system. Inputs are the signals or data received by the system and outputs are the signals or data sent from it. The term can also be used as part of an action; to "perform I/O" is to perform an input or output operation.
I/O devices are the pieces of hardware used by a human (or other system) to communicate with a computer. For instance, a keyboard or computer mouse is an input device for a computer, while monitors and printers are output devices. Devices for communication between computers, such as modems and network cards, typically perform both input and output operations.
The designation of a device as either input or output depends on perspective. Mouse and keyboards take physical movements that the human user outputs and convert them into input signals that a computer can understand; the output from these devices is the computer's input. Similarly, printers and monitors take signals that a computer outputs as input, and they convert these signals into a representation that human users can understand. From the human user's perspective, the process of reading or seeing these representations is receiving output; this type of interaction between computers and humans is studied in the field of human–computer interaction.
In computer architecture, the combination of the CPU and main memory, to which the CPU can read or write directly using individual instructions, is considered the brain of a computer. Any transfer of information to or from the CPU/memory combo, for example by reading data from a disk drive, is considered I/O.

73. Text 13

Embedded system

An embedded system is a controller with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. Embedded systems control many devices in common use today. Ninety-eight percent of all microprocessors manufactured are used in embedded systems.
Modern embedded systems are often based on microcontrollers (i.e. microprocessors with integrated memory and peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more complex systems. In either case, the processor(s) used may be types ranging from general purpose to those specialized in certain class of computations, or even custom designed for the application at hand. A common standard class of dedicated processors is the digital signal processor (DSP). Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.
Embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic light controllers, programmable logic controllers, and large complex systems like hybrid vehicles, medical imaging systems, and avionics. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large equipment rack.

74. Text 14

A microprocessor is a computer processor that incorporates the functions of a central processing unit on a single integrated circuit (IC), or sometimes up to 8 integrated circuits. The microprocessor is a multipurpose, clock driven, register based, digital integrated circuit that accepts binary data as input, processes it according to instructions stored in its memory and provides results (also in binary form) as output. Microprocessors contain both combinational logic and sequential digital logic. Microprocessors operate on numbers and symbols represented in the binary number system.
The integration of a whole CPU onto a single or a few integrated circuits greatly reduced the cost of processing power. Integrated circuit processors are produced in large numbers by highly automated metal-oxide-semiconductor (MOS) fabrication processes, resulting in a low unit price. Single-chip processors increase reliability because there are many fewer electrical connections that could fail. As microprocessor designs improve, the cost of manufacturing a chip (with smaller components built on a semiconductor chip the same size) generally stays the same according to Rock's law.

Before microprocessors, small computers had been built using racks of circuit boards with many medium- and small-scale integrated circuits. Microprocessors combined this into one or a few large-scale ICs. Continued increases in microprocessor capacity have since rendered other forms of computers almost completely obsolete (see history of computing hardware), with one or more microprocessors used in everything from the smallest embedded systems and handheld devices to the largest mainframes and supercomputers.

75. Text 15

BIOS

BIOS is firmware used to perform hardware initialization during the booting process (power-on startup), and to provide runtime services for operating systems and programs. The BIOS firmware comes pre-installed on a personal computer's system board, and it is the first software to run when powered on. The name originates from the Basic Input/Output System used in the CP/M operating system in 1975. The BIOS originally proprietary to the IBM PC has been reverse engineered by companies looking to create compatible systems. The interface of that original system serves as a de facto standard.

The BIOS in modern PCs initializes and tests the system hardware components, and loads a boot loader from a mass memory device which then initializes an operating system. In the era of DOS, the BIOS provided a hardware abstraction layer for the keyboard, display, and other input/output (I/O) devices that standardized an interface to application programs and the operating system. More recent operating systems do not use the BIOS after loading, instead accessing the hardware components directly.

Most BIOS implementations are specifically designed to work with a particular computer or motherboard model, by interfacing with various devices that make up the complementary system chipset. Originally, BIOS firmware was stored in a ROM chip on the PC motherboard. In modern computer systems, the BIOS contents are stored on flash memory so it can be rewritten without removing the chip from the motherboard. This allows easy, end-user updates to the BIOS firmware so new features can be added or bugs can be fixed, but it also creates a possibility for the computer to become infected with BIOS rootkits. Furthermore, a BIOS upgrade that fails can brick the motherboard permanently, unless the system includes some form of backup for this case.

76. Text 1

Supercomputer

A supercomputer is a computer with a high level of performance as compared to a general-purpose computer. The performance of a supercomputer is commonly measured in floating-point operations per second (FLOPS) instead of million instructions per second (MIPS). Since 2017, there are supercomputers which can perform over a hundred quadrillion FLOPS (petaFLOPS). Since November 2017, all of the world's fastest 500 supercomputers run Linux-based operating systems. Additional research is being conducted in China, the United States, the European Union, Taiwan and Japan to build even faster, more powerful and technologically superior exascale supercomputers.

Supercomputers play an important role in the field of computational science, and are used for a wide range of computationally intensive tasks in various fields, including quantum mechanics, weather forecasting, climate research, oil and gas exploration, molecular modeling (computing the structures and properties of chemical compounds, biological macromolecules, polymers, and crystals), and physical simulations (such as simulations of the early moments of the universe, airplane and spacecraft aerodynamics, the detonation of nuclear weapons, and nuclear fusion). Throughout their history, they have been essential in the field of cryptanalysis.

77. Text 2

Virtual machine

In computing, a virtual machine (VM) is an emulation of a computer system. Virtual machines are based on computer architectures and provide functionality of a physical computer. Their implementations may involve specialized hardware, software, or a combination.

There are different kinds of virtual machines, each with different functions:

- System virtual machines (also termed full virtualization VMs) provide a substitute for a real machine. They provide functionality needed to execute entire operating systems. A hypervisor uses native execution to share and manage hardware, allowing for multiple environments which are

isolated from one another, yet exist on the same physical machine. Modern hypervisors use hardware-assisted virtualization, virtualization-specific hardware, primarily from the host CPUs.

- Process virtual machines are designed to execute computer programs in a platform-independent environment.

Some virtual machines, such as QEMU, are designed to also emulate different architectures and allow execution of software applications and operating systems written for another CPU or architecture. Operating-system-level virtualization allows the resources of a computer to be partitioned via the kernel. The terms are not universally interchangeable.

78. Text 3

Flowcharts

Flowcharts are used in designing and documenting simple processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help understand a process, and perhaps also find less-obvious features within the process, like flaws and bottlenecks. There are different types of flowcharts: each type has its own set of boxes and notations. The two most common types of boxes in a flowchart are:

- processing step, usually called activity, and denoted as a rectangular box.
- decision, usually denoted as a diamond.

A flowchart is described as "cross-functional" when the chart is divided into different vertical or horizontal parts, to describe the control of different organizational units. A symbol appearing in a particular part is within the control of that organizational unit. A cross-functional flowchart allows the author to correctly locate the responsibility for performing an action or making a decision, and to show the responsibility of each organizational unit for different parts of a single process.

Flowcharts depict certain aspects of processes and are usually complemented by other types of diagram.

79. Text 4

Pseudocode

Pseudocode is an informal high-level description of the operating principle of a computer program or other algorithm. It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading. Pseudocode typically omits details that are essential for machine understanding of the algorithm, such as variable declarations, system-specific code and some subroutines. The programming language is augmented with natural language description details, where convenient, or with compact mathematical notation. The purpose of using pseudocode is that it is easier for people to understand than conventional programming language code, and that it is an efficient and environment-independent description of the key principles of an algorithm. It is commonly used in textbooks and scientific publications that are documenting various algorithms, and also in planning of computer program development, for sketching out the structure of the program before the actual coding takes place.

No standard for pseudocode syntax exists, as a program in pseudocode is not an executable program.

80. Text 5

Optimizing compiler

In computing, an optimizing compiler is a compiler that tries to minimize or maximize some attributes of an executable computer program. Common requirements are to minimize a program's execution time, memory requirement, and power consumption

Compiler optimization is generally implemented using a sequence of optimizing transformations, algorithms which take a program and transform it to produce a semantically equivalent output program that uses fewer resources and/or executes faster. It has been shown that some code optimization problems are NP-complete (a problem is NP-complete when it can be solved by a restricted class of brute force search), or even undecidable. In practice, factors such as the programmer's willingness to wait for the compiler to complete its task place upper limits on the optimizations that a compiler implementer might provide. (Optimization is generally a very CPU- and memory-intensive process.) In the past, computer memory limitations were also a major factor in limiting which optimizations could be performed. Because of these factors, optimization rarely produces "optimal" output in any sense, and in fact an "optimization" may impede performance in some cases; rather, they are heuristic methods for improving resource usage in typical programs.

81. Text 6

Computer organization

The questions from computer organization usually test the basic knowledge that one acquires at the beginning of computer learning. It also tests the knowledge of candidates on various computer parts and their functioning. The computer system is formed when 2-3 parts combine and perform individually as well as coherently.

Output units allow the computers to send the data to other users. Usually, a display device is considered as the output unit because it displays the texts, graphics, and other information. The common examples of output units are speakers, monitors, printers, etc.

The devices that are used to convey the information to the computer are called input devices. The primary examples of input units are a keyboard, pointing devices, audio/video devices, etc. With the help of input unit, a user can transfer the data to a computer for storing, displaying and processing data. Some of the functions of input devices:

1. Pointing devices:.
2. Keyboard:
3. Audio/video devices:

CPU is known as the brains of the computer. Without CPU a computer cannot work. It allows the computer to interpret and execute the various data through software and hardware. There are three various functioning of CPU. They are:

1.Memory unit
2.Arithmetic-logic unit
3. Control unit

82. Text 7

Motherboard

A motherboard (sometimes alternatively known as the mainboard, main circuit board) is the main printed circuit board (PCB) found in general purpose computers and other expandable systems. It holds, and allows, communication between many of the crucial electronic components of a system, such as the central processing unit (CPU) and memory, and provides connectors for other peripherals. Unlike a backplane, a motherboard usually contains significant sub-systems such as the central processor, the chipset's input/output and memory controllers, interface connectors, and other components integrated for general purpose use and applications.

Motherboard specifically refers to a PCB with expansion capability and as the name suggests, this board is often referred to as the "mother" of all components attached to it, which often include peripherals, interface cards, and daughtercards: sound cards, video cards, network cards, hard drives, or other forms of persistent storage; TV tuner cards, cards providing extra USB or FireWire slots and a variety of other custom components.

Similarly, the term mainboard is applied to devices with a single board and no additional expansions or capability, such as controlling boards in laser printers, televisions, washing machines, mobile phones and other embedded systems with limited expansion abilities.

83. Text 8

Server

In computing, a server is a computer program or a device that provides functionality for other programs or devices, called "clients". This architecture is called the client–server model, and a single overall computation is distributed across multiple processes or devices. Servers can provide various functionalities, often called "services", such as sharing data or resources among multiple clients, or performing computation for a client. A single server can serve multiple clients, and a single client can use multiple servers. A client process may run on the same device or may connect over a network to a server on a different device. Typical servers are database servers, file servers, mail servers, print servers, web servers, game servers, and application servers.

Client–server systems are today most frequently implemented by (and often identified with) the request–response model: a client sends a request to the server, which performs some action and sends a response back to the client, typically with a result or acknowledgement. Designating a computer as "server-class hardware" implies that it is specialized for running servers on it. This often

implies that it is more powerful and reliable than standard personal computers, but alternatively, large computing clusters may be composed of many relatively simple, replaceable server components.

84. Text 9

Laptop

A laptop, often called a notebook, is a small, portable personal computer (PC) with a "clamshell" form factor, typically having a thin LCD or LED computer screen mounted on the inside of the upper lid of the clamshell and an alphanumeric keyboard on the inside of the lower lid. The clamshell is opened up to use the computer. Laptops are folded shut for transportation, and thus are suitable for mobile use. Its name comes from lap, as it was deemed to be placed on a person's lap when being used. Although originally there was a distinction between laptops and notebooks (the former being bigger and heavier than the latter), as of 2014, there is often no longer any difference. Laptops are commonly used in a variety of settings, such as at work, in education, for playing games, Internet surfing, for personal multimedia, and general home computer use.

Laptops combine all the input/output components and capabilities of a desktop computer, including the display screen, small speakers, a keyboard, hard disk drive, optical disc drive, pointing devices (such as a touchpad or trackpad), a processor, and memory into a single unit. Most modern laptops feature integrated webcams and built-in microphones, while many also have touchscreens.

85. Text 10

Low-level programming

A low-level programming language is a programming language that provides little or no abstraction from a computer's instruction set architecture—commands or functions in the language map closely to processor instructions. Generally, this refers to either machine code or assembly language. The word "low" refers to the small or nonexistent amount of abstraction between the language and machine language; because of this, low-level languages are sometimes described as being "close to the hardware". Programs written in low-level languages tend to be relatively non-portable.

Low-level languages can convert to machine code without a compiler or interpreter – second-generation programming languages use a simpler processor called an assembler – and the resulting code runs directly on the processor. A program written in a low-level language can be made to run very quickly, with a small memory footprint. An equivalent program in a high-level language can be less efficient and use more memory. Low-level languages are simple, but considered difficult to use, due to numerous technical details that the programmer must remember.

86. Text 11

Interpreter

In computer science, an interpreter is a computer program that directly executes instructions written in a programming or scripting language, without requiring them previously to have been compiled into a machine language program. An interpreter generally uses one of the following strategies for program execution:

Parse the source code and perform its behavior directly;

Translate source code into some efficient intermediate representation and immediately execute this;

Explicitly execute stored precompiled cod made by a compiler which is part of the interpreter system.

Early versions of Lisp programming language and Dartmouth BASIC would be examples of the first type. Perl, Python, MATLAB, and Ruby are examples of the second, while Pascal is an example of the third type. Source programs are compiled ahead of time and stored as machine independent code, which is then linked at run-time and executed by an interpreter and/or compiler. Some systems, such as Smalltalk and contemporary versions of BASIC and Java may also combine two and three. Interpreters of various types have also been constructed for many languages traditionally associated with compilation, such as Algol, Fortran, Cobol, and C/C++.

87. Text 12

Programming paradigms

Programming paradigms are a way to classify programming languages based on their features. Languages can be classified into multiple paradigms.

Some paradigms are concerned mainly with implications for the execution model of the language, such as allowing side effects, or whether the sequence of operations is defined by the execution model. Other paradigms are concerned mainly with the way that code is organized, such as grouping a code into units along with the state that is modified by the code. Yet others are concerned mainly with the style of syntax and grammar.
Common programming paradigms include:
- imperative in which the programmer instructs the machine how to change its state,
- procedural which groups instructions into procedures,
- object-oriented which groups instructions together with the part of the state they operate on,
- declarative in which the programmer merely declares properties of the desired result, but not how to compute it
- functional in which the desired result is declared as the value of a series of function applications,
- logic in which the desired result is declared as the answer to a question about a system of facts and rules,
- mathematical in which the desired result is declared as the solution of an optimization problem

88. Text 13

Quality assurance

Quality assurance (QA) is a way of preventing mistakes and defects in manufactured products and avoiding problems when delivering products or services to customers; which ISO 9000 defines as "part of quality management focused on providing confidence that quality requirements will be fulfilled". This defect prevention in quality assurance differs subtly from defect detection and rejection in quality control and has been referred to as a shift left since it focuses on quality earlier in the process (i.e., to the left of a linear process diagram reading left to right).
The terms "quality assurance" and "quality control" are often used interchangeably to refer to ways of ensuring the quality of a service or product. For instance, the term "assurance" is often used as follows: Implementation of inspection and structured testing as a measure of quality assurance in a television set software project at Philips Semiconductors is described. The term "control", however, is used to describe the fifth phase of the Define, Measure, Analyze, Improve, Control (DMAIC) model. DMAIC is a data-driven quality strategy used to improve processes.

89. Text 14

Software testing

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.
Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:
- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- it is sufficiently usable,
- can be installed and run in its intended environments, and
- achieves the general result its stakeholders desire.

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors.

90. Text 15

Unit testing

Unit testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to ensure that the building blocks of the software work independently from each other.
Unit testing is a software development process that involves a synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development life cycle. Unit testing aims to eliminate construction errors before code is promoted to additional testing; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development process.

Заочная форма обучения, Первый семестр, Зачет
Контролируемые ИДК: УК-4.1 УК-4.2 УК-4.3
Вопросы/Задания:

1. How worried are you about Internet security?

2. Have you ever had computer problems?

3. What risks are there of being online?

4. What do you think about pop-up windows?

5. What is malware?

6. What do you think about this scam?

7. Could fraudsters one day make the Internet too dangerous?

8. What do you think of the idea of passwords?

9. What do you think of the Web Authentication system?

10. What do you know about biometrics?

11. How secure do you think fingerprints and facial recognition are?

12. Have you ever been hacked?

13. How dangerous is the Internet?

14. What happens when someone steals your identity?

15. What is your biggest worry about being online?

16. What advice do you have for people to stay safe online?

17. What do you know about facial recognition?

18. What are the advantages and disadvantages of facial recognition?

19. Should facial recognition be everywhere?

20. Do you think police officers should wear body cameras?

21. How else can police find criminals quickly?

22. What happens to cities with outdated technology?

23. Would you like to see facial recognition in your town?

24. What do you know about Big Brother?

25. Why are people afraid of the idea of Big Brother?

26. How do we go about making our online transactions secure?

27. What is cloud computing?

28. What is programming?

29. What programming languages do you know?

30. What is flash memory?

31. What is a computer?

32. What is data?

33. What are main functions of a computer?

34. Give examples of using computers in everyday life.

35. What types of computers do you know?

36. What is the role of computers in our sour society?

37. What is a program?

38. What is a computer system?

39. What is a hardware?

40. What is a User Interface?

41. What is information technology?

42. Give the definition of software.

43. Are you a research student?

44. What department are you in?

45. Who Is your scientific adviser (supervisor) of your master's work?

46. What is the subject of your research?

47. What problems does your master's work concentrate on?

48. What problems does your current research include?

49. What methods do you apply in your research? Do you use any new technologies?

50. What web sites do you use for research work?

51. Have you any publications on the subject you study?

52. Where do you work now and as what? Is your work related to computer?

53. What are the research interests of your supervisor?

54. What is the difference between the first computer and the modern?

55. How do computers treat people?

56. How do computers help people?

57. What is the Internet?

58. How can the Internet be useful educational purposes?

59. Why do people want to be "on the internet"?

60. What does a computer virus cause?

Заочная форма обучения, Второй семестр, Экзамен
Контролируемые ИДК: УК-4.1 УК-4.2 УК-4.3
Вопросы/Задания:

1. How worried are you about Internet security?

2. Have you ever had computer problems?

3. What risks are there of being online?

4. What do you think about pop-up windows?

5. What is malware?

6. What do you think about this scam?

7. Could fraudsters one day make the Internet too dangerous?

8. What do you think of the idea of passwords?

9. What do you think of the Web Authentication system?

10. What do you know about biometrics?

11. How secure do you think fingerprints and facial recognition are?

12. Have you ever been hacked?

13. How dangerous is the Internet?

14. What happens when someone steals your identity?

15. What is your biggest worry about being online?

16. What advice do you have for people to stay safe online?

17. What do you know about facial recognition?

18. What are the advantages and disadvantages of facial recognition?

19. Should facial recognition be everywhere?

20. Do you think police officers should wear body cameras?

21. How else can police find criminals quickly?

22. What happens to cities with outdated technology?

23. Would you like to see facial recognition in your town?

24. What do you know about Big Brother?

25. Why are people afraid of the idea of Big Brother?

26. How do computers treat people?

27. How do computers help people?

28. What is the Internet?

29. How can the Internet be useful educational purposes?

30. Why do people want to be “on the internet”?

31. What does a computer virus cause?

32. What dangerous are there for a society which depends on computer screens rather than face-to-face contact for its main means of communications

33. What direction in the magistracy did you choose? Why?

34. Why did you decide to enter the magistracy and in what direction?

35. What should you study more computer science (informatics) or math at your faculty?

36. What does pervasive computing mean?

37. What do you think – How has the computer changed the world?

38. What risks do hackers pose for Internet users?

39. How can one avoid risks using Internet?

40. What way is there to protect a message?

41. What are the most common methods of private networks protection?

42. What are Malwares (Malicious Software)?

43. What is a worm?

44. What is the origin of the word ‘hacker’?

45. Who are ‘white hats’ in computer industry?

46. What password can be considered secure?

47. The Internet guides people to the one, open society, doesn't it? Give examples. What cloud computing services are available?

48. Give examples of cloud computing.

49. What is the history of cloud computing?

50. Who is the hyper scale cloud computing provider?

51. What do abbreviations AT and ML stand for?

52. Why are Al and ML experiencing a strong surge of interest recently’?

53. What language can computers understand?

54. What languages are called ‘low-level’ languages?

55. What languages are called ‘high-level’ languages?

56. Can you name some ‘high-level’ languages?

57. What is a program?

58. What is information technology?

59. What department are you in?

60. What web sites do you use for research work?

61. Text 1

Internet of Things

What if those new jeans you’ve just bought start tweeting about your location as you cross London Bridge? It sounds bizarre, but it’s possible – if they are equipped with a tiny RFID device, your location could be revealed without you knowing about it. This technology is just one of the current ways of allowing physical objects to go online – the so-called Internet of Things. Those in favour of the IoT claim that interconnectivity would allow us to locate and monitor everything, everywhere and at any time. Imagine a smart building where you know how many people are inside just by detecting movement with motion-sensitive lights. This could help save lives in an emergency. But as more objects become part of the digital world, there is growing debate over the benefits of smart technology versus the lack of privacy. To what extent can surveillance of people be accepted? Which principles should govern the use of the IoT? The European Commission, for example, has established a framework to safeguard consumer privacy as industries develop this technology further. Within the retail industry, a number of stores have started using RFID tags to check and track stock more easily. However, some people are worried that the RFID reader being used by a shop employee to check the number of pairs of jeans could also read the data on a customer’s driving license, for example, if it contained a RFID chip. This could then lead to identity theft. If the tag is not removed at the checkout, the item could be tracked on the street. Once the tag is thrown away, it can still be scanned, allowing someone to get an idea of your shopping habits. Supporters of the IoT point out that in our already digital and high-tech society your mobile phone operator and bank know much more about your life than your partner does and it is certainly more critical information than the type of jeans you wear.

62. Text 2

Information

Information can be thought of as the resolution of uncertainty; it is that which answers the question of "what an entity is" and thus defines both its essence and nature of its characteristics. It is associated with data, as data represents values attributed to parameters, and information is data in context and with meaning attached. Information relates also to knowledge, as knowledge signifies understanding of an abstract or concrete concept.

In terms of communication, information is expressed either as the content of a message or through direct or indirect observation. That which is perceived can be construed as a message in its own right, and in that sense, information is always conveyed as the content of a message.

Information can be encoded into various forms for transmission and interpretation (for example, information may be encoded into a sequence of signs, or transmitted via a signal). It can also be encrypted for safe storage and communication.

The uncertainty of an event is measured by its probability of occurrence and is inversely proportional to that. The more uncertain an event, the more information is required to resolve uncertainty of that event. The bit is a typical unit of information, but other units such as the natural unit of information may be used. For example, the information encoded in one "fair" coin flip is log2(2/1) = 1 bit, and in two fair coin flips is log2(4/1) = 2 bits.

The concept of information has different meanings in different contexts. Thus the concept becomes related to notions of constraint, communication, control, data, form, education, knowledge, meaning,

understanding, mental stimuli, pattern, representation, and entropy.

63. Text 3

C++ library

The C++ standard consists of two parts: the core language and the standard library. C++ programmers expect the latter on every major implementation of C++; it includes aggregate types (vectors, lists, maps, sets, queues, stacks, arrays, tuples), algorithms (find, for each, binary search, random shuffle, etc.), input/output facilities (iostream, for reading from and writing to the console and files), filesystem library, localization support, smart pointers for automatic memory management, regular expression support, multi-threading library, atomics support (allowing a variable to be read or written to by at most one thread at a time without any external synchronization), time utilities (measurement, getting current time, etc.), a system for converting error reporting that doesn't use C++ exceptions into C++ exceptions, a random number generator and a slightly modified version of the C standard library (to make it comply with the C++ type system).

A large part of the C++ library is based on the Standard Template Library (STL). Useful tools provided by the STL include containers as the collections of objects (such as vectors and lists), iterators that provide array-like access to containers, and algorithms that perform operations such as searching and sorting.

Furthermore, (multi)maps (associative arrays) and (multi)sets are provided, all of which export compatible interfaces. Therefore, using templates it is possible to write generic algorithms that work with any container or on any sequence defined by iterators. As in C, the features of the library are accessed by using the #include directive to include a standard header. The C++ Standard Library provides 105 standard headers, of which 27 are deprecated.

64. Text 4

Static polymorphism

Function overloading allows programs to declare multiple functions having the same name but with different arguments. The functions are distinguished by the number or types of their formal parameters. Thus, the same function name can refer to different functions depending on the context in which it is used. The type returned by the function is not used to distinguish overloaded functions and would result in a compile-time error message.

When declaring a function, a programmer can specify for one or more parameters a default value. Doing so allows the parameters with defaults to optionally be omitted when the function is called, in which case the default arguments will be used. When a function is called with fewer arguments than there are declared parameters, explicit arguments are matched to parameters in left-to-right order, with any unmatched parameters at the end of the parameter list being assigned their default arguments. In many cases, specifying default arguments in a single function declaration is preferable to providing overloaded function definitions with different numbers of parameters.

Templates in C++ provide a sophisticated mechanism for writing generic, polymorphic code (i.e. parametric polymorphism). In particular, through the curiously recurring template pattern, it's possible to implement a form of static polymorphism that closely mimics the syntax for overriding virtual functions. Because C++ templates are type-aware and Turing-complete, they can also be used to let the compiler resolve recursive conditionals and generate substantial programs through template metaprogramming. Contrary to some opinion, template code will not generate a bulk code after compilation with the proper compiler settings.

65. Text 5

CPU

A central processing unit (CPU), also called a central processor or main processor, is the electronic circuitry within a computer that executes instructions that make up a computer program. The CPU performs basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions. The computer industry has used the term "central processing unit" at least since the early 1960s. Traditionally, the term "CPU" refers to a processor, more specifically to its processing unit and control unit (CU), distinguishing these core elements of a computer from external components such as main memory and I/O circuitry.

The form, design, and implementation of CPUs have changed over the course of their history, but

their fundamental operation remains almost unchanged. Principal components of a CPU include the arithmetic logic unit (ALU) that performs arithmetic and logic operations, processor registers that supply operands to the ALU and store the results of ALU operations, and a control unit that orchestrates the fetching (from memory) and execution of instructions by directing the coordinated operations of the ALU, registers and other components.
Most modern CPUs are microprocessors, where the CPU is contained on a single metal-oxide-semiconductor (MOS) integrated circuit (IC) chip. An IC that contains a CPU may also contain memory, peripheral interfaces, and other components of a computer; such integrated devices are variously called microcontrollers or systems on a chip (SoC). Some computers employ a multi-core processor, which is a single chip containing two or more CPUs called "cores"; in that context, one can speak of such single chips as "sockets".

66. Text 6

Data organization

A modern digital computer represents data using the binary numeral system. Text, numbers, pictures, audio, and nearly any other form of information can be converted into a string of bits, or binary digits, each of which has a value of 1 or 0. The most common unit of storage is the byte, equal to 8 bits. A piece of information can be handled by any computer or device whose storage space is large enough to accommodate the binary representation of the piece of information, or simply data. For example, the complete works of Shakespeare, about 1250 pages in print, can be stored in about five megabytes (40 million bits) with one byte per character.
Data are encoded by assigning a bit pattern to each character, digit, or multimedia object. Many standards exist for encoding
By adding bits to each encoded unit, redundancy allows the computer to both detect errors in coded data and correct them based on mathematical algorithms. Errors generally occur in low probabilities due to random bit value flipping, or "physical bit fatigue", loss of the physical bit in storage of its ability to maintain a distinguishable value (0 or 1), or due to errors in inter or intra-computer communication. A random bit flip (due to random radiation) is typically corrected upon detection. A bit, or a group of malfunctioning physical bits (not always the specific defective bit is known; group definition depends on specific storage device) is typically automatically fenced-out, taken out of use by the device, and replaced with another functioning equivalent group in the device, where the corrected bit values are restored (if possible). The cyclic redundancy check (CRC) method is typically used in communications and storage for error detection. A detected error is then retried.

67. Text 7

Linux-V server

Linux-VServer is a virtual private server implementation that was created by adding operating system-level virtualization capabilities to the Linux kernel. It is developed and distributed as open-source software.
The project was started by Jacques Gélinas. It is now maintained by Herbert Pötzl. It is not related to the Linux Virtual Server project, which implements network load balancing.
Linux-VServer is a jail mechanism in that it can be used to securely partition resources on a computer system (such as the file system, CPU time, network addresses and memory) in such a way that processes cannot mount a denial-of-service attack on anything outside their partition.
Each partition is called a security context, and the virtualized system within it is the virtual private server. A chroot-like utility for descending into security contexts is provided. Booting a virtual private server is then simply a matter of kickstarting init in a new security context; likewise, shutting it down simply entails killing all processes with that security context. The contexts themselves are robust enough to boot many Linux distributions unmodified, including Debian and Fedora.
Virtual private servers are commonly used in web hosting services, where they are useful for segregating customer accounts, pooling resources and containing any potential security breaches. To save space on such installations, each virtual server's file system can be created as a tree of copy-on-write hard links to a "template" file system. The hard link is marked with a special filesystem attribute and when modified, is securely and transparently replaced with a real copy of the file.

68. Text 8

Computer engineering

Computer engineering (CE) is a branch of engineering that integrates several fields of computer science and electronic engineering required to develop computer hardware and software. Computer engineers usually have training in electronic engineering (or electrical engineering), software design, and hardware-software integration instead of only software engineering or electronic engineering. Computer engineers are involved in many hardware and software aspects of computing, from the design of individual microcontrollers, microprocessors, personal computers, and supercomputers, to circuit design. This field of engineering not only focuses on how computer systems themselves work but also how they integrate into the larger picture.

Usual tasks involving computer engineers include writing software and firmware for embedded microcontrollers, designing VLSI chips, designing analog sensors, designing mixed signal circuit boards, and designing operating systems. Computer engineers are also suited for robotics research, which relies heavily on using digital systems to control and monitor electrical systems like motors, communications, and sensors.

In many institutions of higher learning, computer engineering students are allowed to choose areas of in-depth study in their junior and senior year because the full breadth of knowledge used in the design and application of computers is beyond the scope of an undergraduate degree. Other institutions may require engineering students to complete one or two years of general engineering before declaring computer engineering as their primary focus.

69. Text 9

Computer programming

Computer programming is the process of designing and building an executable computer program for accomplishing a specific computing task. Programming involves tasks such as: analysis, generating algorithms, profiling algorithms' accuracy and resource consumption, and the implementation of algorithms in a chosen programming language (commonly referred to as coding). The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit. The purpose of programming is to find a sequence of instructions that will automate the performance of a task (which can be as complex as an operating system) on a computer, often for solving a given problem. The process of programming thus often requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.

Tasks accompanying and related to programming include: testing, debugging, source code maintenance, implementation of build systems, and management of derived artifacts, such as the machine code of computer programs. These might be considered part of the programming process, but often the term software development is used for this larger process with the term programming, implementation, or coding reserved for the actual writing of code. Software engineering combines engineering techniques with software development practices. Reverse engineering is the opposite process. A hacker is any skilled computer expert that uses their technical knowledge to overcome a problem, but it can also mean a security hacker in common language.

70. Text 10

Machine code

Machine code is a computer program written in machine language instructions that can be executed directly by a computer's central processing unit (CPU). Each instruction causes the CPU to perform a very specific task, such as a load, a store, a jump, or an ALU operation on one or more units of data in CPU registers or memory.

Machine code is a strictly numerical language which is intended to run as fast as possible, and may be regarded as the lowest-level representation of a compiled or assembled computer program or as a primitive and hardware-dependent programming language. While it is possible to write programs directly in machine code, it is tedious and error prone to manage individual bits and calculate numerical addresses and constants manually. For this reason, programs are very rarely written directly in machine code in modern contexts, but may be done for low level debugging, program patching (especially when assembler source is not available) and assembly language disassembly.

The overwhelming majority of practical programs today are written in higher-level languages or assembly language. The source code is then translated to executable machine code by utilities such

as compilers, assemblers, and linkers, with the important exception of interpreted programs, which are not translated into machine code. However, the interpreter itself, which may be seen as an executor or processor, performing the instructions of the source code, typically consists of directly executable machine code (generated from assembly or high-level language source code).
Machine code is by definition the lowest level of programming detail visible to the programmer, but internally many processors use microcode or optimize and transform machine code instructions into sequences of micro-ops. This is not generally considered to be a machine code.

71. Text 11

Computer software

Computer software, or simply software, is a collection of data or computer instructions that tell the computer how to work. This is in contrast to physical hardware, from which the system is built and actually performs the work. In computer science and software engineering, computer software is all information processed by computer systems, programs and data. Computer software includes computer programs, libraries and related non-executable data, such as online documentation or digital media. Computer hardware and software require each other and neither can be realistically used on its own.
At the lowest programming level, executable code consists of machine language instructions supported by an individual processor—typically a central processing unit (CPU) or a graphics processing unit (GPU). A machine language consists of groups of binary values signifying processor instructions that change the state of the computer from its preceding state. For example, an instruction may change the value stored in a particular storage location in the computer—an effect that is not directly observable to the user. An instruction may also invoke one of many input or output operations, for example displaying some text on a computer screen; causing state changes which should be visible to the user. The processor executes the instructions in the order they are provided, unless it is instructed to "jump" to a different instruction, or is interrupted by the operating system. As of 2015, most personal computers, smartphone devices and servers have processors with multiple execution units or multiple processors performing computation together, and computing has become a much more concurrent activity than in the past.

72. Text 12

Input/output

In computing, input/output or I/O (or, informally, io or IO) is the communication between an information processing system, such as a computer, and the outside world, possibly a human or another information processing system. Inputs are the signals or data received by the system and outputs are the signals or data sent from it. The term can also be used as part of an action; to "perform I/O" is to perform an input or output operation.
I/O devices are the pieces of hardware used by a human (or other system) to communicate with a computer. For instance, a keyboard or computer mouse is an input device for a computer, while monitors and printers are output devices. Devices for communication between computers, such as modems and network cards, typically perform both input and output operations.
The designation of a device as either input or output depends on perspective. Mouse and keyboards take physical movements that the human user outputs and convert them into input signals that a computer can understand; the output from these devices is the computer's input. Similarly, printers and monitors take signals that a computer outputs as input, and they convert these signals into a representation that human users can understand. From the human user's perspective, the process of reading or seeing these representations is receiving output; this type of interaction between computers and humans is studied in the field of human–computer interaction.
In computer architecture, the combination of the CPU and main memory, to which the CPU can read or write directly using individual instructions, is considered the brain of a computer. Any transfer of information to or from the CPU/memory combo, for example by reading data from a disk drive, is considered I/O.

73. Text 13

Embedded system

An embedded system is a controller with a dedicated function within a larger mechanical or

electrical system, often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. Embedded systems control many devices in common use today. Ninety-eight percent of all microprocessors manufactured are used in embedded systems.

Modern embedded systems are often based on microcontrollers (i.e. microprocessors with integrated memory and peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more complex systems. In either case, the processor(s) used may be types ranging from general purpose to those specialized in certain class of computations, or even custom designed for the application at hand. A common standard class of dedicated processors is the digital signal processor (DSP). Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic light controllers, programmable logic controllers, and large complex systems like hybrid vehicles, medical imaging systems, and avionics. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large equipment rack.

74. Text 14

A microprocessor is a computer processor that incorporates the functions of a central processing unit on a single integrated circuit (IC), or sometimes up to 8 integrated circuits. The microprocessor is a multipurpose, clock driven, register based, digital integrated circuit that accepts binary data as input, processes it according to instructions stored in its memory and provides results (also in binary form) as output. Microprocessors contain both combinational logic and sequential digital logic. Microprocessors operate on numbers and symbols represented in the binary number system.

The integration of a whole CPU onto a single or a few integrated circuits greatly reduced the cost of processing power. Integrated circuit processors are produced in large numbers by highly automated metal-oxide-semiconductor (MOS) fabrication processes, resulting in a low unit price. Single-chip processors increase reliability because there are many fewer electrical connections that could fail. As microprocessor designs improve, the cost of manufacturing a chip (with smaller components built on a semiconductor chip the same size) generally stays the same according to Rock's law.

Before microprocessors, small computers had been built using racks of circuit boards with many medium- and small-scale integrated circuits. Microprocessors combined this into one or a few large-scale ICs. Continued increases in microprocessor capacity have since rendered other forms of computers almost completely obsolete (see history of computing hardware), with one or more microprocessors used in everything from the smallest embedded systems and handheld devices to the largest mainframes and supercomputers.

75. Text 15

BIOS

BIOS is firmware used to perform hardware initialization during the booting process (power-on startup), and to provide runtime services for operating systems and programs. The BIOS firmware comes pre-installed on a personal computer's system board, and it is the first software to run when powered on. The name originates from the Basic Input/Output System used in the CP/M operating system in 1975. The BIOS originally proprietary to the IBM PC has been reverse engineered by companies looking to create compatible systems. The interface of that original system serves as a de facto standard.

The BIOS in modern PCs initializes and tests the system hardware components, and loads a boot loader from a mass memory device which then initializes an operating system. In the era of DOS, the BIOS provided a hardware abstraction layer for the keyboard, display, and other input/output (I/O) devices that standardized an interface to application programs and the operating system. More recent operating systems do not use the BIOS after loading, instead accessing the hardware components directly.

Most BIOS implementations are specifically designed to work with a particular computer or motherboard model, by interfacing with various devices that make up the complementary system

chipset. Originally, BIOS firmware was stored in a ROM chip on the PC motherboard. In modern computer systems, the BIOS contents are stored on flash memory so it can be rewritten without removing the chip from the motherboard. This allows easy, end-user updates to the BIOS firmware so new features can be added or bugs can be fixed, but it also creates a possibility for the computer to become infected with BIOS rootkits. Furthermore, a BIOS upgrade that fails can brick the motherboard permanently, unless the system includes some form of backup for this case.

76. Text 1

Supercomputer

A supercomputer is a computer with a high level of performance as compared to a general-purpose computer. The performance of a supercomputer is commonly measured in floating-point operations per second (FLOPS) instead of million instructions per second (MIPS). Since 2017, there are supercomputers which can perform over a hundred quadrillion FLOPS (petaFLOPS). Since November 2017, all of the world's fastest 500 supercomputers run Linux-based operating systems. Additional research is being conducted in China, the United States, the European Union, Taiwan and Japan to build even faster, more powerful and technologically superior exascale supercomputers.

Supercomputers play an important role in the field of computational science, and are used for a wide range of computationally intensive tasks in various fields, including quantum mechanics, weather forecasting, climate research, oil and gas exploration, molecular modeling (computing the structures and properties of chemical compounds, biological macromolecules, polymers, and crystals), and physical simulations (such as simulations of the early moments of the universe, airplane and spacecraft aerodynamics, the detonation of nuclear weapons, and nuclear fusion). Throughout their history, they have been essential in the field of cryptanalysis.

77. Text 2

Virtual machine

In computing, a virtual machine (VM) is an emulation of a computer system. Virtual machines are based on computer architectures and provide functionality of a physical computer. Their implementations may involve specialized hardware, software, or a combination.

There are different kinds of virtual machines, each with different functions:

- System virtual machines (also termed full virtualization VMs) provide a substitute for a real machine. They provide functionality needed to execute entire operating systems. A hypervisor uses native execution to share and manage hardware, allowing for multiple environments which are isolated from one another, yet exist on the same physical machine. Modern hypervisors use hardware-assisted virtualization, virtualization-specific hardware, primarily from the host CPUs.
- Process virtual machines are designed to execute computer programs in a platform-independent environment.

Some virtual machines, such as QEMU, are designed to also emulate different architectures and allow execution of software applications and operating systems written for another CPU or architecture. Operating-system-level virtualization allows the resources of a computer to be partitioned via the kernel. The terms are not universally interchangeable.

78. Text 3

Flowcharts

Flowcharts are used in designing and documenting simple processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help understand a process, and perhaps also find less-obvious features within the process, like flaws and bottlenecks. There are different types of flowcharts: each type has its own set of boxes and notations. The two most common types of boxes in a flowchart are:

- processing step, usually called activity, and denoted as a rectangular box.
- decision, usually denoted as a diamond.

A flowchart is described as "cross-functional" when the chart is divided into different vertical or horizontal parts, to describe the control of different organizational units. A symbol appearing in a particular part is within the control of that organizational unit. A cross-functional flowchart allows the author to correctly locate the responsibility for performing an action or making a decision, and to show the responsibility of each organizational unit for different parts of a single process.

Flowcharts depict certain aspects of processes and are usually complemented by other types of

diagram.

79. Text 4

Pseudocode

Pseudocode is an informal high-level description of the operating principle of a computer program or other algorithm. It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading. Pseudocode typically omits details that are essential for machine understanding of the algorithm, such as variable declarations, system-specific code and some subroutines. The programming language is augmented with natural language description details, where convenient, or with compact mathematical notation. The purpose of using pseudocode is that it is easier for people to understand than conventional programming language code, and that it is an efficient and environment-independent description of the key principles of an algorithm. It is commonly used in textbooks and scientific publications that are documenting various algorithms, and also in planning of computer program development, for sketching out the structure of the program before the actual coding takes place.

No standard for pseudocode syntax exists, as a program in pseudocode is not an executable program.

80. Text 5

Optimizing compiler

In computing, an optimizing compiler is a compiler that tries to minimize or maximize some attributes of an executable computer program. Common requirements are to minimize a program's execution time, memory requirement, and power consumption

Compiler optimization is generally implemented using a sequence of optimizing transformations, algorithms which take a program and transform it to produce a semantically equivalent output program that uses fewer resources and/or executes faster. It has been shown that some code optimization problems are NP-complete (a problem is NP-complete when it can be solved by a restricted class of brute force search), or even undecidable. In practice, factors such as the programmer's willingness to wait for the compiler to complete its task place upper limits on the optimizations that a compiler implementer might provide. (Optimization is generally a very CPU- and memory-intensive process.) In the past, computer memory limitations were also a major factor in limiting which optimizations could be performed. Because of these factors, optimization rarely produces "optimal" output in any sense, and in fact an "optimization" may impede performance in some cases; rather, they are heuristic methods for improving resource usage in typical programs.

81. Text 6

Computer organization

The questions from computer organization usually test the basic knowledge that one acquires at the beginning of computer learning. It also tests the knowledge of candidates on various computer parts and their functioning. The computer system is formed when 2-3 parts combine and perform individually as well as coherently.

Output units allow the computers to send the data to other users. Usually, a display device is considered as the output unit because it displays the texts, graphics, and other information. The common examples of output units are speakers, monitors, printers, etc.

The devices that are used to convey the information to the computer are called input devices. The primary examples of input units are a keyboard, pointing devices, audio/video devices, etc. With the help of input unit, a user can transfer the data to a computer for storing, displaying and processing data. Some of the functions of input devices:

1. Pointing devices:.
2. Keyboard:
3. Audio/video devices:

CPU is known as the brains of the computer. Without CPU a computer cannot work. It allows the computer to interpret and execute the various data through software and hardware. There are three various functioning of CPU. They are:

1.Memory unit
2.Arithmetic-logic unit
3. Control unit

82. Text 7

Motherboard

A motherboard (sometimes alternatively known as the mainboard, main circuit board) is the main printed circuit board (PCB) found in general purpose computers and other expandable systems. It holds, and allows, communication between many of the crucial electronic components of a system, such as the central processing unit (CPU) and memory, and provides connectors for other peripherals. Unlike a backplane, a motherboard usually contains significant sub-systems such as the central processor, the chipset's input/output and memory controllers, interface connectors, and other components integrated for general purpose use and applications.

Motherboard specifically refers to a PCB with expansion capability and as the name suggests, this board is often referred to as the "mother" of all components attached to it, which often include peripherals, interface cards, and daughtercards: sound cards, video cards, network cards, hard drives, or other forms of persistent storage; TV tuner cards, cards providing extra USB or FireWire slots and a variety of other custom components.

Similarly, the term mainboard is applied to devices with a single board and no additional expansions or capability, such as controlling boards in laser printers, televisions, washing machines, mobile phones and other embedded systems with limited expansion abilities.

83. Text 8

Server

In computing, a server is a computer program or a device that provides functionality for other programs or devices, called "clients". This architecture is called the client–server model, and a single overall computation is distributed across multiple processes or devices. Servers can provide various functionalities, often called "services", such as sharing data or resources among multiple clients, or performing computation for a client. A single server can serve multiple clients, and a single client can use multiple servers. A client process may run on the same device or may connect over a network to a server on a different device. Typical servers are database servers, file servers, mail servers, print servers, web servers, game servers, and application servers.

Client–server systems are today most frequently implemented by (and often identified with) the request–response model: a client sends a request to the server, which performs some action and sends a response back to the client, typically with a result or acknowledgement. Designating a computer as "server-class hardware" implies that it is specialized for running servers on it. This often implies that it is more powerful and reliable than standard personal computers, but alternatively, large computing clusters may be composed of many relatively simple, replaceable server components.

84. Text 9

Laptop

A laptop, often called a notebook, is a small, portable personal computer (PC) with a "clamshell" form factor, typically having a thin LCD or LED computer screen mounted on the inside of the upper lid of the clamshell and an alphanumeric keyboard on the inside of the lower lid. The clamshell is opened up to use the computer. Laptops are folded shut for transportation, and thus are suitable for mobile use. Its name comes from lap, as it was deemed to be placed on a person's lap when being used. Although originally there was a distinction between laptops and notebooks (the former being bigger and heavier than the latter), as of 2014, there is often no longer any difference. Laptops are commonly used in a variety of settings, such as at work, in education, for playing games, Internet surfing, for personal multimedia, and general home computer use.

Laptops combine all the input/output components and capabilities of a desktop computer, including the display screen, small speakers, a keyboard, hard disk drive, optical disc drive, pointing devices (such as a touchpad or trackpad), a processor, and memory into a single unit. Most modern laptops feature integrated webcams and built-in microphones, while many also have touchscreens.

85. Text 10

Low-level programming

A low-level programming language is a programming language that provides little or no abstraction from a computer's instruction set architecture—commands or functions in the language map closely

to processor instructions. Generally, this refers to either machine code or assembly language. The word "low" refers to the small or nonexistent amount of abstraction between the language and machine language; because of this, low-level languages are sometimes described as being "close to the hardware". Programs written in low-level languages tend to be relatively non-portable.
Low-level languages can convert to machine code without a compiler or interpreter – second-generation programming languages use a simpler processor called an assembler – and the resulting code runs directly on the processor. A program written in a low-level language can be made to run very quickly, with a small memory footprint. An equivalent program in a high-level language can be less efficient and use more memory. Low-level languages are simple, but considered difficult to use, due to numerous technical details that the programmer must remember.

86. Text 11

Interpreter

In computer science, an interpreter is a computer program that directly executes instructions written in a programming or scripting language, without requiring them previously to have been compiled into a machine language program. An interpreter generally uses one of the following strategies for program execution:
Parse the source code and perform its behavior directly;
Translate source code into some efficient intermediate representation and immediately execute this;
Explicitly execute stored precompiled cod made by a compiler which is part of the interpreter system.
Early versions of Lisp programming language and Dartmouth BASIC would be examples of the first type. Perl, Python, MATLAB, and Ruby are examples of the second, while Pascal is an example of the third type. Source programs are compiled ahead of time and stored as machine independent code, which is then linked at run-time and executed by an interpreter and/or compiler. Some systems, such as Smalltalk and contemporary versions of BASIC and Java may also combine two and three. Interpreters of various types have also been constructed for many languages traditionally associated with compilation, such as Algol, Fortran, Cobol, and C/C++.

87. Text 12

Programming paradigms

Programming paradigms are a way to classify programming languages based on their features. Languages can be classified into multiple paradigms.
Some paradigms are concerned mainly with implications for the execution model of the language, such as allowing side effects, or whether the sequence of operations is defined by the execution model. Other paradigms are concerned mainly with the way that code is organized, such as grouping a code into units along with the state that is modified by the code. Yet others are concerned mainly with the style of syntax and grammar.
Common programming paradigms include:
- imperative in which the programmer instructs the machine how to change its state,
- procedural which groups instructions into procedures,
- object-oriented which groups instructions together with the part of the state they operate on,
- declarative in which the programmer merely declares properties of the desired result, but not how to compute it
- functional in which the desired result is declared as the value of a series of function applications,
- logic in which the desired result is declared as the answer to a question about a system of facts and rules,
- mathematical in which the desired result is declared as the solution of an optimization problem

88. Text 13

Quality assurance

Quality assurance (QA) is a way of preventing mistakes and defects in manufactured products and avoiding problems when delivering products or services to customers; which ISO 9000 defines as "part of quality management focused on providing confidence that quality requirements will be fulfilled". This defect prevention in quality assurance differs subtly from defect detection and rejection in quality control and has been referred to as a shift left since it focuses on quality earlier in the process (i.e., to the left of a linear process diagram reading left to right).

The terms "quality assurance" and "quality control" are often used interchangeably to refer to ways of ensuring the quality of a service or product. For instance, the term "assurance" is often used as follows: Implementation of inspection and structured testing as a measure of quality assurance in a television set software project at Philips Semiconductors is described. The term "control", however, is used to describe the fifth phase of the Define, Measure, Analyze, Improve, Control (DMAIC) model. DMAIC is a data-driven quality strategy used to improve processes.

89. Text 14

Software testing

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- it is sufficiently usable,
- can be installed and run in its intended environments, and
- achieves the general result its stakeholders desire.

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors.

90. Text 15

Unit testing

Unit testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to ensure that the building blocks of the software work independently from each other.

Unit testing is a software development process that involves a synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development life cycle. Unit testing aims to eliminate construction errors before code is promoted to additional testing; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development process.

8. Материально-техническое и учебно-методическое обеспечение дисциплины

8.1. Перечень основной и дополнительной учебной литературы

Основная литература

1. СТЕПАНОВА А.П Деловой английский язык: учеб. пособие / СТЕПАНОВА А.П, Погребняк Н.В.. - Краснодар: КубГАУ, 2019. - 81 с. - 978-5-00097-856-6. - Текст: непосредственный.

2. Лисковец,, И. В. Иностранный язык (английский язык). Сборник упражнений по переводу и разговорной речи: учебное пособие / И. В. Лисковец,, Ю. В. Смирнова,. - Иностранный язык (английский язык). Сборник упражнений по переводу и разговорной речи - Санкт-Петербург: Санкт-Петербургский государственный университет промышленных технологий и дизайна, 2019. - 112 c. - 2227-8397. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/102418.html (дата обращения: 08.10.2025). - Режим доступа: по подписке

3. Данилова,, Л. Р. Английский язык: учебное пособие / Л. Р. Данилова,, Е. А. Горбаренко,; под редакцией Л. Р. Данилова. - Английский язык - Санкт-Петербург: Санкт-Петербургский государственный архитектурно-строительный университет, ЭБС АСВ, 2017. - 136 c. - 978-5-9227-0748-0. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/78589.html (дата обращения: 08.10.2025). - Режим доступа: по подписке

Дополнительная литература

1. Попов,, Е. Б. Miscellaneous items. Общеразговорный английский язык: учебное пособие / Е. Б. Попов,. - Miscellaneous items. Общеразговорный английский язык - Саратов: Вузовское образование, 2019. - 132 c. - 978-5-4487-0457-4. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/79610.html (дата обращения: 08.09.2025). - Режим доступа: по подписке

2. Английский язык для неязыковых факультетов: учебник / сост. А. Д. Караулова. - Английский язык для неязыковых факультетов - Астрахань: Астраханский государственный архитектурно-строительный университет, ЭБС АСВ, 2019. - 128 c. - 978-5-93026-057-1. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/93075.html (дата обращения: 08.10.2025). - Режим доступа: по подписке

3. Караулова,, А. Д. Английский язык для неязыковых факультетов. Ч.2: учебное пособие / А. Д. Караулова,. - Английский язык для неязыковых факультетов. Ч.2 - Астрахань: Астраханский государственный архитектурно-строительный университет, ЭБС АСВ, 2021. - 121 c. - 978-5-93026-144-8. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/123429.html (дата обращения: 08.10.2025). - Режим доступа: по подписке

4. Нестеренко,, В. Г. Информативное чтение на английском языке: учебно-методическое пособие для студентов i и ii курсов неязыковых специальностей / В. Г. Нестеренко,. - Информативное чтение на английском языке - Саратов: Вузовское образование, 2018. - 49 c. - 978-5-4487-0013-2. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/76828.html (дата обращения: 08.09.2025). - Режим доступа: по подписке

5. КАРИПИДИ А. Г. Иностранный язык (английский): метод. указания / КАРИПИДИ А. Г.. - Краснодар: КубГАУ, 2020. - 32 с. - Текст: электронный. // : [сайт]. - URL: https://edu.kubsau.ru/mod/resource/view.php?id=7878 (дата обращения: 12.05.2026). - Режим доступа: по подписке

6. КОЧКИНА В. П. Деловой английский язык неличные формы глагола: учеб. пособие / КОЧКИНА В. П., Степанова А. П.. - Краснодар: КубГАУ, 2019. - 117 с. - 978-5-907294-08-0. - Текст: электронный. // : [сайт]. - URL: https://edu.kubsau.ru/mod/resource/view.php?id=7009 (дата обращения: 12.05.2026). - Режим доступа: по подписке

8.2. Профессиональные базы данных и ресурсы «Интернет», к которым обеспечивается доступ обучающихся

Профессиональные базы данных

Не используются.

Ресурсы «Интернет»

1. https://znanium.com/
- Znanium.com
2. https://edu.kubsau.ru/ - Образовательный портал КубГАУ
3. http://www.iprbookshop.ru/ - Электронный библлиотечный ресурс
4. http://e.lanbook.com/ - Электронный библиотечный рксурс
5. http://elibrary.rsl.ru/ - Электронная библиотека Российской государственной библиотеки
6. http://elibrary.ru/defaultx.asp - Научная электронная библиотека
7. http://www.lingvo-online.ru/ru - ABBYY Lingvo Live
8. http://www. iqlib.ru - Электронная библиотека IQlib
9. https://lingualeo.com/ru - Lingualeo иностранные языки онлайн
10. https://www.multitran.ru/ - Словарь «Мультитран»
11. http://www.minfin.ru/ru/accounting/mej_standart_fo/docs - Минфин России: Документы МСФО:

8.3. Программное обеспечение и информационно-справочные системы, используемые при осуществлении образовательного процесса по дисциплине

Информационные технологии, используемые при осуществлении образовательного процесса по дисциплине позволяют:
– обеспечить взаимодействие между участниками образовательного процесса, в том числе синхронное и (или) асинхронное взаимодействие посредством сети «Интернет»;
– фиксировать ход образовательного процесса, результатов промежуточной аттестации по дисциплине и результатов освоения образовательной программы;
– организовать процесс образования путем визуализации изучаемой информации посредством использования презентаций, учебных фильмов;
– контролировать результаты обучения на основе компьютерного тестирования.
Перечень лицензионного программного обеспечения:
1 Microsoft Windows - операционная система.
2 Microsoft Office (включает Word, Excel, Power Point) - пакет офисных приложений.
Перечень профессиональных баз данных и информационных справочных систем:
1 Гарант - правовая, https://www.garant.ru/
2 Консультант - правовая, https://www.consultant.ru/
3 Научная электронная библиотека eLibrary - универсальная, https://elibrary.ru/
Доступ к сети Интернет, доступ в электронную информационно-образовательную среду университета.

Перечень программного обеспечения
(обновление производится по мере появления новых версий программы)
Не используется.

Перечень информационно-справочных систем
(обновление выполняется еженедельно)
Не используется.

8.4. Специальные помещения, лаборатории и лабораторное оборудование

Компьютерный класс
226гл
Интерактивная панель Samsung - 1 шт.
Персональный компьютер HP 6300 Pro SFF/Core i3-3220/4GB/500GB/NoODD/Win7Pro - 1 шт.
Сплит-система LS-H12KPA2/LU-H12KPA2 - 1 шт.

Учебная аудитория
308зоо
доска ДК11Э2010 - 1 шт.
доска интерактивная SMART 680 iv - 1 шт.
доска классная - 1 шт.
доска магнитно-маркерная - 1 шт.
доска марк. PREMIUM LEGAMASTER 100×150 - 1 шт.
жалюзи вертикальные - 1 шт.
Магнитола CD/MP3,дека, FM тюнер - 1 шт.
ноутбук HP ProBook 4530s 15.6" - 1 шт.
парты - 1 шт.
Сплит-система LS-H18KPA2/LU-H18KPA2 - 1 шт.
стелаж - 1 шт.
Шкаф для документов - 2 шт.
шкаф платяной - 1 шт.
350зоо
Доска классная - 1 шт.
доска марк. PREMIUM LEGAMASTER 100×150 - 1 шт.
Облучатель-рециркулятор воздуха 600 - 1 шт.
Парты - 15 шт.
стул твердый - 2 шт.
Шкаф книжный - 1 шт.
шкаф комбинированный - 1 шт.
шкаф плотяной - 1 шт.
424зоо
Вешалка для одежды - 1 шт.
доска марк. PREMIUM LEGAMASTER 100×150 - 1 шт.
Магнитола CD/MP3,дека, FM тюнер - 1 шт.
парты - 9 шт.
стол однотумбовый - 1 шт.
Стул мягкий черный - 1 шт.
стул твердый - 1 шт.
шкаф книжный - 1 шт.
шкаф комбинированный - 1 шт.

9. Методические указания по освоению дисциплины (модуля)

Учебная работа по направлению подготовки осуществляется в форме контактной работы с преподавателем, самостоятельной работы обучающегося, текущей и промежуточной аттестаций, иных формах, предлагаемых университетом. Учебный материал дисциплины структурирован и его изучение производится в тематической последовательности. Содержание методических указаний должно соответствовать требованиям Федерального государственного образовательного стандарта и учебных программ по дисциплине. Самостоятельная работа студентов может быть выполнена с помощью материалов, размещенных на портале поддержки Moodl.

Методические указания по формам работы

Практические занятия

Форма организации обучения, проводимая под руководством преподавателя и служащая для детализации, анализа, расширения, углубления, закрепления, применения (или выполнения разнообразных практических работ, упражнений) и контроля усвоения полученной на лекциях учебной информации. Практические занятия проводятся с использованием

учебно-методических изданий, размещенных на образовательном портале университета.

Описание возможностей изучения дисциплины лицами с ОВЗ и инвалидами

Для инвалидов и лиц с ОВЗ может изменяться объём дисциплины (модуля) в часах, выделенных на контактную работу обучающегося с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающегося (при этом не увеличивается количество зачётных единиц, выделенных на освоение дисциплины).

Фонды оценочных средств адаптируются к ограничениям здоровья и восприятия информации обучающимися.

Основные формы представления оценочных средств – в печатной форме или в форме электронного документа.

Формы контроля и оценки результатов обучения инвалидов и лиц с ОВЗ с нарушением зрения:

– устная проверка: дискуссии, тренинги, круглые столы, собеседования, устные коллоквиумы и др.;

– с использованием компьютера и специального ПО: работа с электронными образовательными ресурсами, тестирование, рефераты, курсовые проекты, дистанционные формы, если позволяет острота зрения - графические работы и др.;

– при возможности письменная проверка с использованием рельефно-точечной системы Брайля, увеличенного шрифта, использование специальных технических средств (тифлотехнических средств): контрольные, графические работы, тестирование, домашние задания, эссе, отчеты и др.

Формы контроля и оценки результатов обучения инвалидов и лиц с ОВЗ с нарушением слуха:

– письменная проверка: контрольные, графические работы, тестирование, домашние задания, эссе, письменные коллоквиумы, отчеты и др.;

– с использованием компьютера: работа с электронными образовательными ресурсами, тестирование, рефераты, курсовые проекты, графические работы, дистанционные формы и др.;

– при возможности устная проверка с использованием специальных технических средств (аудиосредств, средств коммуникации, звукоусиливающей аппаратуры и др.): дискуссии, тренинги, круглые столы, собеседования, устные коллоквиумы и др.

Формы контроля и оценки результатов обучения инвалидов и лиц с ОВЗ с нарушением опорно-двигательного аппарата:

– письменная проверка с использованием специальных технических средств (альтернативных средств ввода, управления компьютером и др.): контрольные, графические работы, тестирование, домашние задания, эссе, письменные коллоквиумы, отчеты и др.;

– устная проверка, с использованием специальных технических средств (средств коммуникаций): дискуссии, тренинги, круглые столы, собеседования, устные коллоквиумы и др.;

– с использованием компьютера и специального ПО (альтернативных средств ввода и управления компьютером и др.): работа с электронными образовательными ресурсами, тестирование, рефераты, курсовые проекты, графические работы, дистанционные формы предпочтительнее обучающимся, ограниченным в передвижении и др.

Адаптация процедуры проведения промежуточной аттестации для инвалидов и лиц с ОВЗ.

В ходе проведения промежуточной аттестации предусмотрено:

– предъявление обучающимся печатных и (или) электронных материалов в формах, адаптированных к ограничениям их здоровья;

– возможность пользоваться индивидуальными устройствами и средствами, позволяющими адаптировать материалы, осуществлять приём и передачу информации с учетом их индивидуальных особенностей;

– увеличение продолжительности проведения аттестации;

– возможность присутствия ассистента и оказания им необходимой помощи (занять рабочее место, передвигаться, прочитать и оформить задание, общаться с преподавателем).

Формы промежуточной аттестации для инвалидов и лиц с ОВЗ должны учитывать

индивидуальные и психофизические особенности обучающегося/обучающихся по АОПОП ВО (устно, письменно на бумаге, письменно на компьютере, в форме тестирования и т.п.).
Специальные условия, обеспечиваемые в процессе преподавания дисциплины студентам с нарушениями зрения:
– предоставление образовательного контента в текстовом электронном формате, позволяющем переводить плоскопечатную информацию в аудиальную или тактильную форму;
– возможность использовать индивидуальные устройства и средства, позволяющие адаптировать материалы, осуществлять приём и передачу информации с учетом индивидуальных особенностей и состояния здоровья студента;
– предоставление возможности предкурсового ознакомления с содержанием учебной дисциплины и материалом по курсу за счёт размещения информации на корпоративном образовательном портале;
– использование чёткого и увеличенного по размеру шрифта и графических объектов в мультимедийных презентациях;
– использование инструментов «лупа», «прожектор» при работе с интерактивной доской;
– озвучивание визуальной информации, представленной обучающимся в ходе занятий;
– обеспечение раздаточным материалом, дублирующим информацию, выводимую на экран;
– наличие подписей и описания у всех используемых в процессе обучения рисунков и иных графических объектов, что даёт возможность перевести письменный текст в аудиальный;
– обеспечение особого речевого режима преподавания: лекции читаются громко, разборчиво, отчётливо, с паузами между смысловыми блоками информации, обеспечивается интонирование, повторение, акцентирование, профилактика рассеивания внимания;
– минимизация внешнего шума и обеспечение спокойной аудиальной обстановки;
– возможность вести запись учебной информации студентами в удобной для них форме (аудиально, аудиовизуально, на ноутбуке, в виде пометок в заранее подготовленном тексте);
– увеличение доли методов социальной стимуляции (обращение внимания, апелляция к ограничениям по времени, контактные виды работ, групповые задания и др.) на практических и лабораторных занятиях;
– минимизирование заданий, требующих активного использования зрительной памяти и зрительного внимания;
– применение поэтапной системы контроля, более частый контроль выполнения заданий для самостоятельной работы.
Специальные условия, обеспечиваемые в процессе преподавания дисциплины студентам с нарушениями опорно-двигательного аппарата (маломобильные студенты, студенты, имеющие трудности передвижения и патологию верхних конечностей):
– возможность использовать специальное программное обеспечение и специальное оборудование и позволяющее компенсировать двигательное нарушение (коляски, ходунки, трости и др.);
– предоставление возможности предкурсового ознакомления с содержанием учебной дисциплины и материалом по курсу за счёт размещения информации на корпоративном образовательном портале;
– применение дополнительных средств активизации процессов запоминания и повторения;
– опора на определенные и точные понятия;
– использование для иллюстрации конкретных примеров;
– применение вопросов для мониторинга понимания;
– разделение изучаемого материала на небольшие логические блоки;
– увеличение доли конкретного материала и соблюдение принципа от простого к сложному при объяснении материала;
– наличие чёткой системы и алгоритма организации самостоятельных работ и проверки заданий с обязательной корректировкой и комментариями;
– увеличение доли методов социальной стимуляции (обращение внимания, аппеляция к ограничениям по времени, контактные виды работ, групповые задания др.);
– обеспечение беспрепятственного доступа в помещения, а также пребывания них;
– наличие возможности использовать индивидуальные устройства и средства, позволяющие обеспечить реализацию эргономических принципов и комфортное пребывание на месте в

течение всего периода учёбы (подставки, специальные подушки и др.).
Специальные условия, обеспечиваемые в процессе преподавания дисциплины студентам с нарушениями слуха (глухие, слабослышащие, позднооглохшие):
– предоставление образовательного контента в текстовом электронном формате, позволяющем переводить аудиальную форму лекции в плоскопечатную информацию;
– наличие возможности использовать индивидуальные звукоусиливающие устройства и сурдотехнические средства, позволяющие осуществлять приём и передачу информации; осуществлять взаимообратный перевод текстовых и аудиофайлов (блокнот для речевого ввода), а также запись и воспроизведение зрительной информации;
– наличие системы заданий, обеспечивающих систематизацию вербального материала, его схематизацию, перевод в таблицы, схемы, опорные тексты, глоссарий;
– наличие наглядного сопровождения изучаемого материала (структурно-логические схемы, таблицы, графики, концентрирующие и обобщающие информацию, опорные конспекты, раздаточный материал);
– наличие чёткой системы и алгоритма организации самостоятельных работ и проверки заданий с обязательной корректировкой и комментариями;
– обеспечение практики опережающего чтения, когда студенты заранее знакомятся с материалом и выделяют незнакомые и непонятные слова и фрагменты;
– особый речевой режим работы (отказ от длинных фраз и сложных предложений, хорошая артикуляция; четкость изложения, отсутствие лишних слов; повторение фраз без изменения слов и порядка их следования; обеспечение зрительного контакта во время говорения и чуть более медленного темпа речи, использование естественных жестов и мимики);
– чёткое соблюдение алгоритма занятия и заданий для самостоятельной работы (называние темы, постановка цели, сообщение и запись плана, выделение основных понятий и методов их изучения, указание видов деятельности студентов и способов проверки усвоения материала, словарная работа);
– соблюдение требований к предъявляемым учебным текстам (разбивка текста на час¬ти; выделение опорных смысловых пунктов; использование наглядных средств);
– минимизация внешних шумов;
– предоставление возможности соотносить вербальный и графический материал; комплексное использование письменных и устных средств коммуникации при работе в группе;
– сочетание на занятиях всех видов речевой деятельности (говорения, слушания, чтения, письма, зрительного восприятия с лица говорящего).
Специальные условия, обеспечиваемые в процессе преподавания дисциплины студентам с прочими видами нарушений (ДЦП с нарушениями речи, заболевания эндокринной, центральной нервной и сердечно-сосудистой систем, онкологические заболевания):
– наличие возможности использовать индивидуальные устройства и средства, позволяющие осуществлять приём и передачу информации;
– наличие системы заданий, обеспечивающих систематизацию вербального материала, его схематизацию, перевод в таблицы, схемы, опорные тексты, глоссарий;
– наличие наглядного сопровождения изучаемого материала;
– наличие чёткой системы и алгоритма организации самостоятельных работ и проверки заданий с обязательной корректировкой и комментариями;
– обеспечение практики опережающего чтения, когда студенты заранее знакомятся с материалом и выделяют незнакомые и непонятные слова и фрагменты;
– предоставление возможности соотносить вербальный и графический материал; комплексное использование письменных и устных средств коммуникации при работе в группе;
– сочетание на занятиях всех видов речевой деятельности (говорения, слушания, чтения, письма, зрительного восприятия с лица говорящего);
– предоставление образовательного контента в текстовом электронном формате;
– предоставление возможности предкурсового ознакомления с содержанием учебной дисциплины и материалом по курсу за счёт размещения информации на корпоративном образовательном портале;
– возможность вести запись учебной информации студентами в удобной для них форме (аудиально, аудиовизуально, в виде пометок в заранее подготовленном тексте);

– применение поэтапной системы контроля, более частый контроль выполнения заданий для самостоятельной работы;
– стимулирование выработки у студентов навыков самоорганизации и самоконтроля;
– наличие пауз для отдыха и смены видов деятельности по ходу занятия.

10. Методические рекомендации по освоению дисциплины (модуля)

Учебная работа по направлению подготовки осуществляется в форме контактной работы с преподавателем, самостоятельной работы обучающегося, текущей и промежуточной аттестаций, иных формах, предлагаемых университетом. Учебный материал дисциплины структурирован и его изучение производится в тематической последовательности. Содержание методических рекомендаций должно соответствовать требованиям Федерального государственного образовательного стандарта и учебных программ по дисциплине. Самостоятельная работа студентов может быть выполнена с помощью материалов, размещенных на портале поддержки Moodl.