МИНОБРНАУКИ РОССИИ
Юго-Западный государственный университет

УТВЕРЖДАЮ:
Заведующий кафедрой
космического приборостроения и систем связи
_____ В.Г. Андронов
(подпись)
«30» 08 2024 г.

ОЦЕНОЧНЫЕ СРЕДСТВА
для текущего контроля успеваемости и промежуточной аттестации обучающихся по дисциплине

Основы программирования в инфокоммуникациях
(наименование дисциплины)

11.03.02 Инфокоммуникационные технологии и системы связи, направленность (профиль) «Сети связи и системы коммутации»
(код и наименование ОПОП ВО)

Курск – 2024

1 ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ

1.1 ВОПРОСЫ ДЛЯ СОБЕСЕДОВАНИЯ

Раздел 1. Язык С

1. Сфера применения языка Си.
2. Назовите основные типы данных языка Си.
3. Дайте описание основных свойств типа int.
4. Дайте описание основных свойств типа char.
5. Дайте описание основных свойств типа float.
6. Дайте описание основных свойств типа double.
7. Что такое модификаторы основных типов данных?
8. Каковы возможности языка при работе с массивами?
9. Как в языке Си осуществляется работа с текстовыми данными?
10. Существуют ли логические переменные в языке Си?
11. Приведите примеры арифметических операций в языке Си.
12. Приведите примеры логических операций в языке Си.
13. Что такое операции инкремента в языке Си?
14. Что такое операции декремента в языке Си?
15. Существуют ли операторы ввода-вывода в языке Си?
16. Опишите оператор цикла while.
17. Опишите оператор цикла do-while.
18. Опишите оператор цикла for.
19. Допустимо ли использование оператора goto в языке Си?
20. Охарактеризуйте роль функций в языке Си
21. Опишите разновидности функций, используемых в процессе разработки ПО.
22. Как задается начальная точка исполнения программы в языке Си?
23. Что такое «структура» в языке Си?
23. Что такое «объединение в языке» Си?
24. Опишите назначение и роль структур
25. Что такое указатель в языке Си?
26. Опишите роль указателей в языке Си
27. Что такое локальные переменные в языке Си?
28. Что такое глобальные переменные в языке Си?
29. Как осуществить ввод данных в языке Си?
30. Как осуществить вывод данных в языке Си?
31. Опишите функцию форматированного ввода данных в языке Си
32. Опишите функцию форматированного вывода данных в языке Си
33. Охарактеризуйте возможности языка Си при работе с последовательными файлами.
34. Приведите примеры функций для вывода данных в файл.
35. Приведите примеры функций для ввода данных из файла.
36. Как отследить завершение данных в последовательном файле?
37. Опишите понятие «открыть» файл.
38. Опишите понятие «закрыть» файл.
39. Какие режимы открытия файла Вам известны?
40. Приведите примеры среды программирования на языке Си
41. В чем основные отличия между функциями printf() и puts()?
42. Можно ли вводить значения для нескольких переменных в рамках одного вызова scanf()?
43. Можно ли выводить значения для нескольких переменных в рамках одного вызова printf()?

Раздел 2. Аппаратно-программные платформы инфокоммуникационных систем.

1. Концепция платформы Arduino.
2. Что входит в состав платформы Arduino?
3. Какая модель микро-ЭВМ лежит в основе платформы Arduino?
4. Охарактеризуйте микро-ЭВМ ATMega8.

5. Охарактеризуйте микро-ЭВМ ATMega328.
6. Охарактеризуйте микро-ЭВМ ATMega32U4.
7. Какие виды сигналов предусмотрены для обработки в платформе Arduino?
8. Дайте краткое описание цифровых линий платформы Arduino UNO.
9. Дайте краткое описание аналоговых линий платформы Arduino UNO.
10. Опишите способы загрузки ПО в память команд Arduino.
11. Что представляет собой среда программирования IDE Arduino?
12. Каковы этапы развертывания IDE Arduino на персональном компьютере?
13. Какие возможности обеспечивает среда программирования Arduino?
14. Какие средства отладки предусмотрены в IDE Arduino?
15. Что лежит в основе языка программирования IDE Arduino?
16. Какова структура программы в IDE Arduino?
17. Опишите назначение секции setup.
18. Опишите назначение секции loop.
19. Каким образом осуществляется настройка цифровых линий Arduino?
20. Каким образом осуществляется настройка аналоговых линий Arduino?
21. Каким образом осуществляется вывод информации по цифровым линиям Arduino?
22. Каким образом осуществляется ввод информации с цифровых линий Arduino?
23. Опишите оператор цикла while.
24. Опишите оператор цикла do-while.
25. Опишите оператор цикла for.
26. Опишите способы ветвления в программе на языке IDE Arduino.
27. Опишите способы работы с временными интервалами в IDE Arduino.
28. Каким образом осуществляется ввод информации с аналоговых линий Arduino?
29. Каким образом осуществляется использование последовательного канала (UART) в Arduino?
30. Опишите функции работы со случайными числами в IDE Arduino.
31. Концепция платформы Raspberry Pi.
32. Что входит в состав платформы Raspberry Pi?
33. Какая модель процессора лежит в основе платформы Raspberry Pi?
34. Каковы этапы развертывания операционной системы для платформы Raspberry Pi?
35. Какие операционные системы доступны для платформы Raspberry Pi?
36. Охарактеризуйте операционную систему(ОС) Raspbian.
37. Какие языки программирования доступны для ОС Raspbian?
38. Дайте общее описание языка Python.
39. Какие средства разработки на языке Си доступны для платформы Raspberry Pi?
40. Какие разновидности / модификации / «клоны» Raspberry Pi Вам известны?

Шкала оценивания: 8 балльная.

Критерии оценивания *(нижеследующие критерии оценки являются примерными и могут корректироваться)**:***

7-8 баллов (или оценка **«отлично»)** выставляется обучающемуся, если он принимает активное участие в беседе по большинству обсуждаемых вопросов (в том числе самых сложных); демонстрирует сформированную способность к диалогическому мышлению, проявляет уважение и интерес к иным мнениям; владеет глубокими (в том числе дополнительными) знаниями по существу обсуждаемых вопросов, ораторскими способностями и правилами ведения полемики; строит логичные, аргументированные, точные и лаконичные высказывания, сопровождаемые яркими примерами; легко и заинтересованно откликается на неожиданные ракурсы беседы; не нуждается в уточняющих и (или) дополнительных вопросах преподавателя.

5-6 баллов (или оценка **«хорошо»)** выставляется обучающемуся, если он принимает участие в обсуждении не менее 50% дискуссионных вопросов; проявляет уважение и интерес к иным мнениям, доказательно и корректно защищает свое мнение; владеет хорошими знаниями вопросов, в обсуждении которых принимает участие; умеет не столько вести полемику, сколько участвовать в ней; строит логичные,

аргументированные высказывания, сопровождаемые подходящими примерами; не всегда откликается на неожиданные ракурсы беседы; не нуждается в уточняющих и (или) дополнительных вопросах преподавателя.

4 балла (или оценка **«удовлетворительно»)** выставляется обучающемуся, если он принимает участие в беседе по одному-двум наиболее простым обсуждаемым вопросам; корректно выслушивает иные мнения; неуверенно ориентируется в содержании обсуждаемых вопросов, порой допуская ошибки; в полемике предпочитает занимать позицию заинтересованного слушателя; строит краткие, но в целом логичные высказывания, сопровождаемые наиболее очевидными примерами; теряется при возникновении неожиданных ракурсов беседы и в этом случае нуждается в уточняющих и (или) дополнительных вопросах преподавателя.

0-3 баллов (или оценка **«неудовлетворительно»)** выставляется обучающемуся, если он не владеет содержанием обсуждаемых вопросов или допускает грубые ошибки; пассивен в обмене мнениями или вообще не участвует в дискуссии; затрудняется в построении монологического высказывания и (или) допускает ошибочные высказывания; постоянно нуждается в уточняющих и (или) дополнительных вопросах преподавателя.

1.2 ВОПРОСЫ И ЗАДАНИЯ В ТЕСТОВОЙ ФОРМЕ

Раздел 1. Язык С

1 Вопрос в закрытой форме

1.1 В языке программирования Си в записи выражения int prop[3][4][6];
а) Ошибка вследствие применения ключевого слова
б) Нет ошибок
в) Ошибка вследствие применения служебных символов

1.2 В языке программирования Си в записи выражения int i, *ip, (*ipp)[6]:
а) Ошибка вследствие применения ключевого слова
б) Нет ошибок
в) Ошибка вследствие применения служебных символов

1.3 В языке программирования Си в записи a=123; присутствует:
а) Логическая операция
б) Операция присваивания
в) Арифметическая операция

1.4 В языке программирования Си в записи a+=123; присутствует:
а) Арифметическая операция
б) Запись ошибочна
в) Логическая операция

1.5 В языке программирования Си в записи a/=123; присутствует:
а) Логическая операция
б) Арифметическая операция
в) Запись ошибочна

1.6 В языке программирования Си в записи a-=123; присутствует:
а) Логическая операция
б) Арифметическая операция
в) Запись ошибочна

1.7 В языке программирования Си в записи a&=123; присутствует:
а) Логическая операция И

б) Побитовая операция И
в) Запись ошибочна

1.8 В языке программирования Си в записи a=b & 0xF; присутствует:
а) Запись ошибочна
б) Побитовая операция И
в) Логическая операция

1.9 В языке программирования Си в записи (a==b && b=3) присутствует:
а) Запись ошибочна
б) Логическая операция И
в) Побитовая операция И

1.10 В языке программирования Си в выражения инкремента a++ и ++a:
а) Всегда эквивалентны при использовании
б) Не всегда эквивалентны при использовании
в) Применимы только в языке Си++

1.11
```
int   i, j, k;
i=j+34;
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Ошибка во второй строке
б) Нет синтаксических ошибок
в) Ошибка в третьей строке

1.12
```
int   i, j, k;
i=j+34
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка во второй строке
в) Ошибка в третьей строке

1.13
```
integer i, j, k;
i=j+34;
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в первой строке
в) Ошибка в третьей строке

1.14
```
double i, j, k;
i=j+.34;
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка во второй строке

1.15

```
double int, jnt, knt;
knt=jnt+.34;
knt/= 2.;
```

В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в первой строке
в) Ошибки во второй и третьей строках

1.16

```
int   i; j; k;
i=j+34;
k+= i;
```

В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в первой строке
в) Ошибки во второй и третьей строках

1.17

```
float f1;
int j,k;
j=(int)f1;
```

В приведенном фрагменте программы (рисунок):
а) Ошибка в первой строке
б) Нет синтаксических ошибок
в) Ошибки во второй и третьей строках

1.18

```
float f1;
int j;k;
j=(int)f1;
```

В приведенном фрагменте программы (рисунок):
а) Ошибки в третьей строке
б) Ошибка во второй строке
в) Нет синтаксических ошибок

1.19

```
float f1;
int j,k
j=(int)f1;
```

В приведенном фрагменте программы (рисунок):
а) Ошибки во второй и третьей строках
б) Ошибка во второй строке
в) Нет синтаксических ошибок

1.20

```
float f1;
int j,k;
if(j==k);;
```

В приведенном фрагменте программы (рисунок):
а) Ошибка в первой строке
б) Нет синтаксических ошибок
в) Ошибки в третьей строке

2 Вопрос в открытой форме.

2.1 _____ _____– это совокупность следующих объектов: директив, указаний компилятору, объявлений и определений.

2.2 ____ – это язык программирования общего назначения, хорошо известный своей эффективностью, экономичностью, и переносимостью.

2.3 _____ _____– это последовательность букв, цифр и символов, заключенная в двойные кавычки. _____

2.4 Строчные литералы имеют тип _____.

2.5 _____ – это имена переменных, функций и меток, используемых в программе.

2.6 _____ – это единица текста программы, которая имеет определенный смысл для компилятора и которая не может быть разбита в дальнейшем.

2.7 _____ _____ – это изменения состояния машины, которые возникают в результате вычисления выражений.

2.8 Си не поддерживает оператор "else if", но тот же самый эффект достигается посредством сложенных операторов _____.

2.9 _____ – это независимая совокупность объявлений и операторов, обычно предназначенная для выполнения определенной задачи.

2.10 Программы на Си состоят по крайней мере из одной функции ____, но могут содержать и больше функций.

3 Вопросы на установление соответствия.

3.1 Установить соответствие основных операторов языка С с их описанием:

1) `if-else`	a) Циклический оператор, который выполняет определенный блок кода до тех пор, пока условие истинно.
2) `for`	b) Оператор выбора, который позволяет выбирать выполнение кода в зависимости от значения выражения.
3) `while`	c) Оператор выбора, который позволяет выполнить различный набор инструкций в зависимости от условия.
4) `switch`	d) Циклический оператор, используемый для выполнения определенного блока кода заданное количество раз.

3.2 Установить соответствие типов данных в языке С с их описанием:

1) `int`	a) Целочисленный тип данных, который используется для хранения целых чисел.
2) `char`	b) Тип данных, предназначенный для хранения символов, таких как буквы и знаки пунктуации.
3) `float`	c) Тип данных с плавающей точкой, предназначенный для хранения чисел с плавающей запятой одинарной точности.
4) `double`	d) Тип данных с плавающей точкой двойной точности, предназначенный для хранения чисел с высокой точностью.

3.3 Установить соответствие:

1) Массив	a) Упорядоченный набор данных одного типа, которые могут быть доступны по индексу.

2) Строка (массив символов)	b) Массив символов, который представляет собой последовательность символов, завершающуюся нулевым символом.
3) Индекс	c) Число, которое используется для указания позиции элемента в массиве или строке.
4) Элемент	d) Конкретное значение в массиве или строке, находящееся на определенной позиции.

3.4 Установить соответствие:

1) `if`	a) Оператор выбора, который выполняет код, если условие истинно, иначе переходит к другой ветке.
2) `else`	b) Оператор, который выполняется, если условие в старшем операторе ложно.
3) `switch`	c) Конструкция выбора, которая позволяет выбирать выполнение кода в зависимости от значения выражения.
4) `? :` (`тернарный` оператор)	d) Тернарный оператор, который выполняет одно из двух выражений в зависимости от значения условия.

3.5 Установить соответствие:

1) `fopen()`	a) Функция, которая открывает файл для чтения или записи и возвращает указатель на файл.
2) `fclose()`	b) Функция, которая закрывает открытый файл и освобождает ресурсы.
3) `fread()`	c) Функция, которая считывает данные из файла и сохраняет их в буфере.
4) `fwrite()`	d) Функция, которая записывает данные из буфера в файл.

3.6 Установить соответствие:

1) print()	a) Используется для вывода текста или данных на экран.
2) input()	b) Используется для получения пользовательского ввода с клавиатуры.
3) open()	c) Используется для открытия файла в режиме чтения или записи.
4) close()	d) Используется для закрытия открытого файла.

3.7 Установить соответствие:

1) `printf()`	a) Функция, используемая для вывода данных на консоль.
2) `scanf()`	b) Функция, используемая для чтения данных с консоли.
3) `stdlib.h`	c) Библиотека, которая содержит функции для работы с памятью и другими стандартными функциями.
4) `string.h`	d) Библиотека, которая содержит функции для работы со строками, включая копирование, сравнение и другие операции.

3.8 Установить соответствие:

1) `malloc()`	a) Функция, которая заполняет блок памяти указанным значением.
2) `free()`	b) Функция, которая копирует данные из одного блока памяти в другой.
3) `memset()`	c) Функция, которая используется для выделения блока динамической памяти заданного размера в куче.
4) `memcpy()`	d) Функция, которая освобождает ранее выделенную динамическую память.

3.9 Установить соответствие:

1) `#include`	a) Директива, которая включает содержимое указанного файла в исходный код программы.
2) `#define`	b) Директива, которая используется для создания макросов и замены их в коде программы.
`3) #ifdef`	c) Директива, которая проверяет, определен ли макрос препроцессора, и выполняет код в зависимости от результата.
`4) #ifndef`	d) Директива, которая проверяет, не определен ли макрос препроцессора, и выполняет код в зависимости от результата.

3.10 Установить соответствие:

1) `errno`	a) Глобальная переменная, которая содержит код ошибки после выполнения некоторых системных функций.
2) `perror()`	b) Функция, которая выводит сообщение об ошибке, соответствующее текущему значению `errno`.
3) `try-catch`	c) Конструкция, используемая для обработки исключительных ситуаций и ошибок в программе.
4) `setjmp()` и `longjmp()`	d) Функции, которые позволяют реализовать механизм низкоуровневой обработки исключений в C.

4 Вопросы на установление последовательности

4.1 Упорядочьте следующие действия в правильном порядке для вывода чисел от 1 до 10 с использованием цикла for:

а) Условие продолжения выполнения цикла.
б) Инкремент переменной счетчика.
в) Вывод текущего значения счетчика.
г) Инициализация переменной счетчика.

4.2 Расставьте этапы компиляции программы на языке C в правильном порядке:

а) Препроцессинг.
б) Компиляция.
в) Ассемблирование.
г) Линковка.

4.3 Установите правильный порядок этапов разработки приложения для мобильного устройства:

а) Определение функциональных требований и интерфейса приложения.
б) Написание и отладка программного кода.
в) Тестирование на различных устройствах и ОС.

г) Подготовка к публикации в магазине приложений.
д) Проектирование пользовательского интерфейса.

4.4 Установите последовательность выполнения операторов в языке С:
а) Условные операторы (if, else).
б) Циклы (for, while).
в) Операции присваивания.
г) Арифметические операции.

4.5 В какой последовательности происходит процесс линковки программы на Си?
а) Обработка символов
б) Разрешение внешних ссылок
в) Подключение стандартных библиотек
г) Подключение пользовательских библиотек

4.6 В каком порядке вызываются функции стандартной библиотеки языка Си при выполнении программы?
main()
а) Математические функции
б) Функции ввода/вывода
в) Функции обработки строк
г) Функции управления памятью

4.7 В каком порядке происходит создание и уничтожение объектов в программе на Си с использованием конструкторов и деструкторов?
а) Создание объектов
б) Вызов конструкторов
в) Вызов деструкторов
г) Уничтожение объектов

4.8 Установите последовательность выполнения цикла программы на Arduino:
а) Инициализация
б) Выполнение
в) Проверка условия
г) Обновление переменных

4.9 В каком порядке располагаются элементы в структуре данных в языке Си?
а) Определение структуры
б) Инициализация структуры
в) Доступ к элементам структуры
г) Модификация элементов структуры
д) Уничтожение структуры

4.10 Установите последовательность выполнения задачи на Raspberry Pi с использованием Python:
а) Импорт необходимых модулей
б) Запуск основного кода
в) Обработка данных или выполнение задачи
г) Завершение

4.11 Перетащите и упорядочьте следующие шаги в правильной последовательности:
а) Написание исходного кода программы.

б) Запуск компилятора (например, GCC) с указанием имени исходного файла.
в) Проверка исходного кода на наличие синтаксических ошибок.
г) Сохранение программы с расширением .c.
д) Анализ вывода компилятора на наличие ошибок компиляции.
е) Выполнение скомпилированной программы.

Раздел 2. Аппаратно-программные платформы инфокоммуникационных систем
1 Вопрос в закрытой форме.
1.1

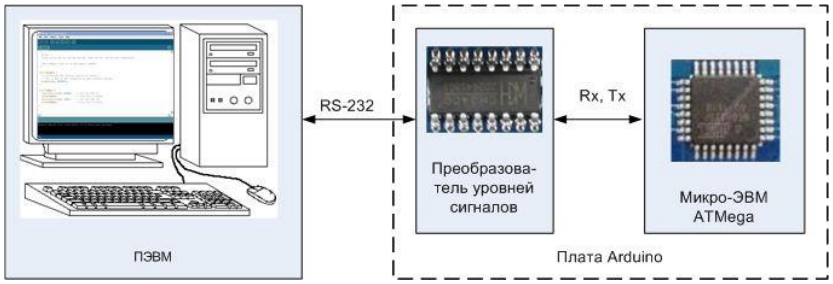
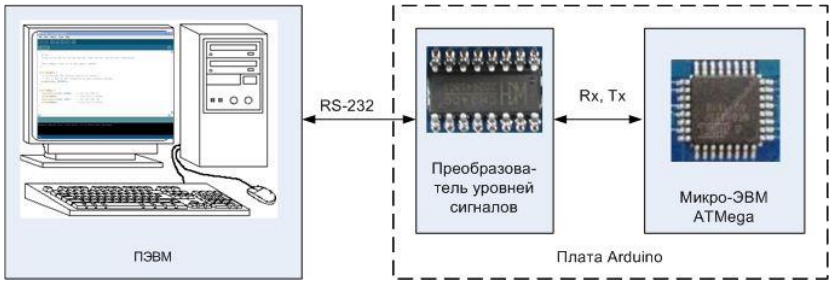


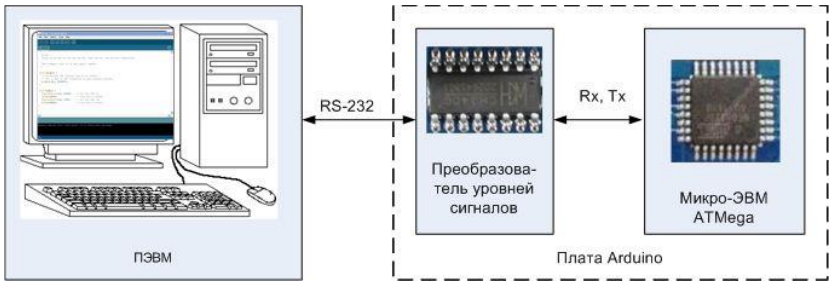
Рисунок иллюстрирует:
а) соединение для загрузки программного кода в платформу Arduino
б) все перечисленные варианты правильные
в) соединение для обмена данными с ПЭВМ в процессе работы системы на базе Arduino

1.2



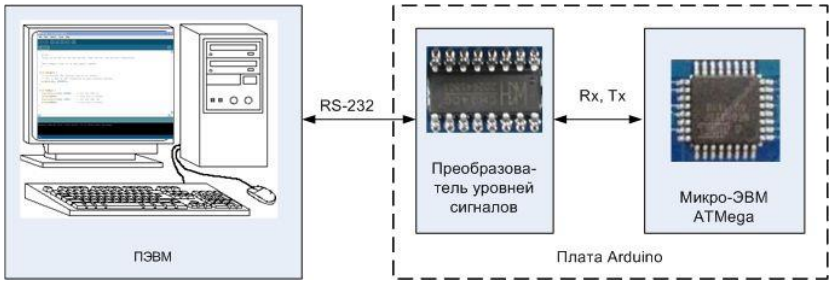
На рисунке показано:
а) загрузка программного кода из ПЭВМ по интерфейсу USB
б) загрузка программного кода из ПЭВМ по интерфейсу RS-232
в) обмен данными между ПЭВМ и Arduino по интерфейсу USB

1.3



Преобразователь уровней сигналов на рисунке:
а) преобразует как уровни сигналов, так и поток данных из ПЭВМ
б) преобразует только уровни сигналов для согласования
в) устанавливается изготовителем на элитных платах исключительно с коммерческой целью

1.4



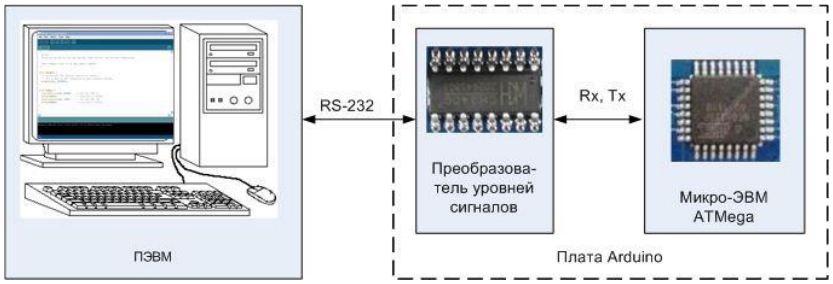
Среда разработки для платформы Arduino (рисунок):

а) включает специальное программное обеспечение для ПЭВМ, коммутационные средства для соединения и не требует предустановки содержимого памяти команд микро-ЭВМ

б) включает специальное программное обеспечение для ПЭВМ, предустановленное содержимое памяти команд микро-ЭВМ и коммутационные средства для соединения

в) включает предустановленное содержимое памяти команд микро-ЭВМ, коммутационные средства для соединения и использует стандартные средства ОС Windows

1.5



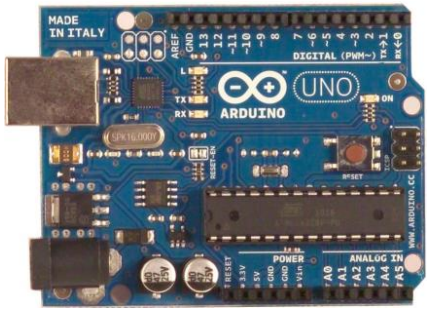
Специальное программное обеспечение на ПЭВМ для среды Arduino выполняет функции:

а) только для трансляции пользовательских программ в машинный код и передачи их на плату Arduino

б) трансляции пользовательских программ в машинный код, передачи их на плату Arduino, запуска записанной программы и при необходимости - обмена данными с платформой Arduino

в) трансляции пользовательских программ в машинный код, передачи их на плату Arduino, запуска записанной программы и её динамической отладки в процессе исполнения

1.6



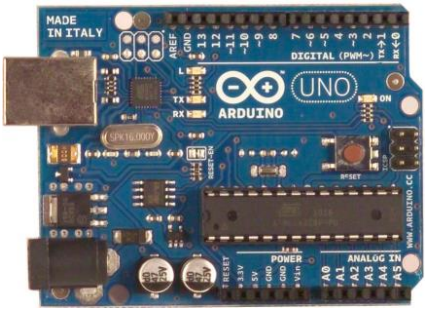
Среда разработки программ Arduino основана:

а) на языке Pascal

б) на языке C

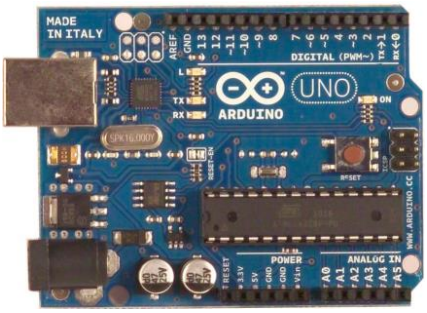
в) на собственном языке Arduino

1.7



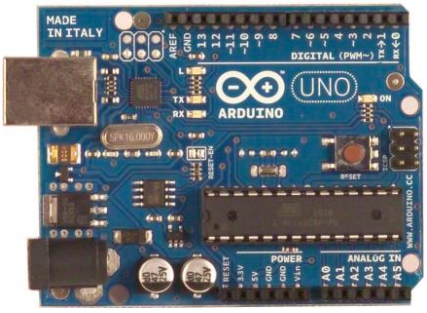
Структурно программа (sketch) для среды Arduino содержит:
а) функцию main() и пользовательские функции
б) функции setup(), loop(), пользовательские функции
в) только пользовательские функции

1.8



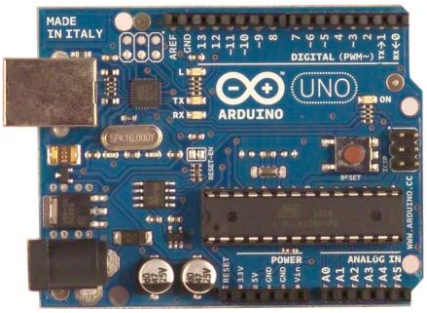
Переменные в программе (sketch) могут быть определены:
а) только внутри функций setup() и loop()
б) в любом структурном подмодуле программы
в) только в начале программы, до функции setup()

1.9



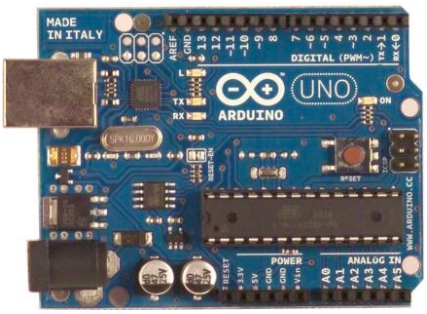
Функция setup() в программе (sketch) Arduino:
а) вызывается при необходимости из функции loop()
б) выполняется первой, до вызова остальных функций
в) порядок выполнения функций задается программистом

1.10



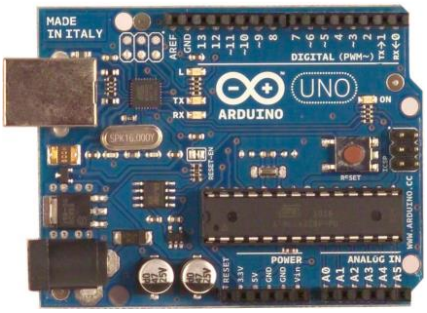
Функция loop в программе (sketch):
а) вызывается при необходимости из функции setup()
б) образует бесконечный цикл
в) вызывается однократно

1.11



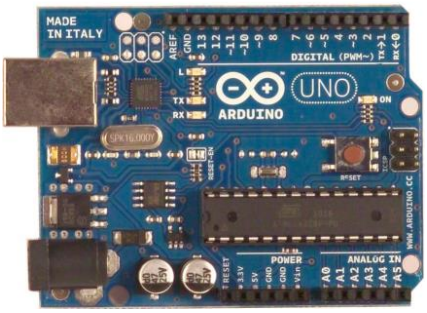
В программе (sketch) Arduino:
а) пользовательские функции могут быть определены только внутри структурной единицы loop()
б) имеется возможность создавать и использовать пользовательские функции
в) допускается использовать только библиотечные функции

1.12



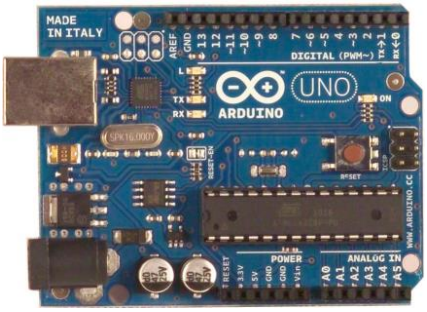
Основное назначение функции setup() в программе (sketch) Arduino:
а) объявление и задание начальных значений всех переменных
б) обеспечение необходимых настроек оборудования на нужные режимы работы
в) запуск механизма загрузки программного кода в память команд микро-ЭВМ платы Arduino

1.13



Основное назначение функции loop() в программе (sketch) Arduino:
а) эта функция однократно запускает заданный программистом алгоритм
б) организация бесконечного цикла для повторения заданного программистом алгоритма
в) эта функция предназначена для осуществления отладки программы и позволяет прервать процесс исполнения при возникновении определенных условий

1.14



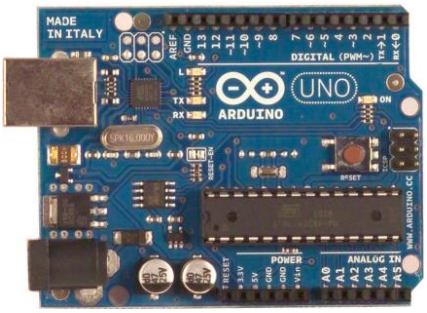
Программа (sketch) Arduino по эффективности (быстродействию и объему кода):

а) существенно лучше программных продуктов, написанным на языке Си и ассемблер, так как оптимизирована именно для Arduino

б) потенциально уступает продуктам, написанным на языке Си и ассемблер

в) все перечисленные варианты равнозначны

1.15



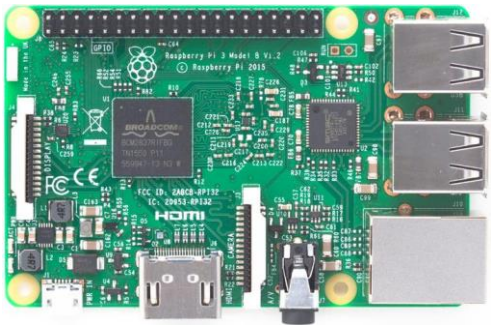
Программа (sketch) Arduino при формировании исполнимого (машинного) кода:

а) это интерпретируемый текст, наподобие Basic или Javaб) образует бесконечный цикл

б) преобразуется в текст на языке Си, а затем компилируется

в) сразу компилируется в машинные коды специальным компилятором с языка Arduino

1.16



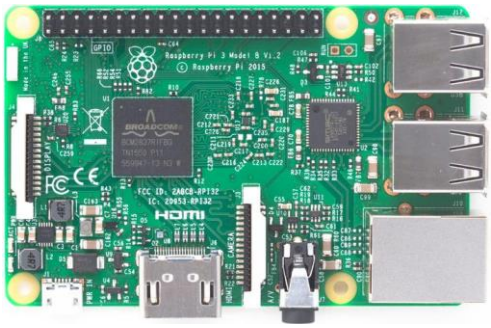
Платформа Raspberry Pi представляет собой:

а) Arduino-подобную платформу, предназначенную для управления объектами, но имеющую более высокую производительность

б) миниатюрный компьютер, к которому можно подключить стандартные монитор, клавиатуру и мышь

в) приставку к персональному компьютеру, предназначенную для расширенной передачи управляющих сигналов управляемым объектам

1.17



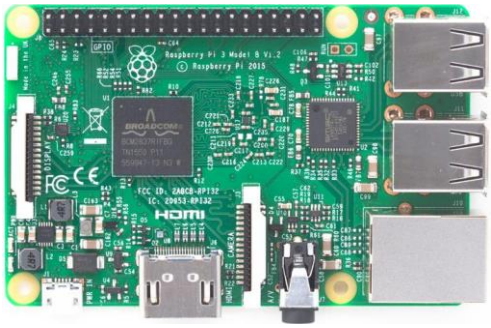
Программное обеспечение платформы Raspberry Pi базируется на операционной системе:

а) как и в Arduino, применение операционной системы не предусмотрено

б) основанной на ОС Linux

в) Microsoft Windows XP или 7

1.18



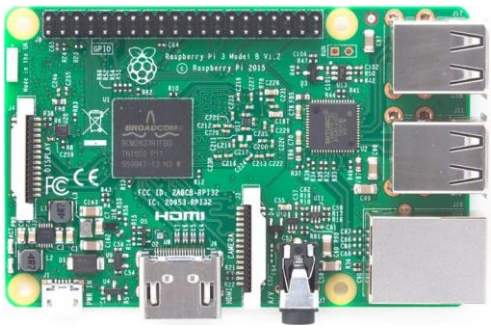
Обладает ли платформа Raspberry Pi (до модели Pi 3 включительно) встроенными средствами для управления электронными объектами?

а) да, включая цифровые линии ввода-вывода, интерфейсы I2C, SPI, RS-232, а также линии аналогового ввода-вывода

б) да, включая цифровые линии ввода-вывода и интерфейсы I2C, SPI, RS-232

в) нет, как и персональный компьютер. Но можно использовать оборудование, подключаемое посредством USB

1.19



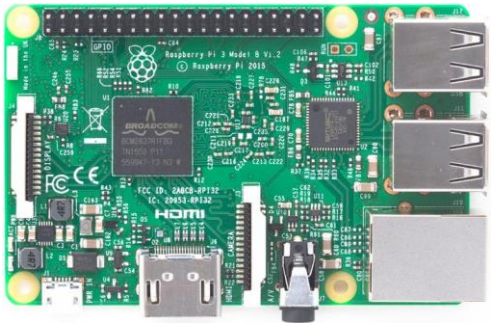
Основной сферой использования Raspberry Pi следует считать:

а) замену платформ Arduino в тех разработках, где требуется большая вычислительная производительность или функциональность

б) всё перечисленное

в) применение в разнообразных разработках электронных изделий энтузиастами, не являющимися профессиональными программистами

1.20



Для разработки программ на платформе Raspberry Pi:

а) как и для Arduino, требуется дополнительный персональный компьютер, с которого и будет происходить загрузка программы для исполнения

б) достаточно возможностей самой платформы

в) средства разработки ограничены, используют только набор готовых программ, предоставляемый разработчиком платформы

2 Вопрос в открытой форме.

2.1_____ – это идентификатор, который может быть определен с квадратными скобками ([]), звездочкой (*) или круглыми скобками () для объявления массива, указателя или функции.

2.2 Спецификаторы типов float и double относятся к типу "_____".

2.3 _____ _____ – это свойство, позволяющее выполнять программы на разных ЭВМ, работающих под управлением разных версий ОС UNIX, с минимальными изменениями.

2.4 1TBS является аббревиатурой стиля _____.

2.5 В среде программирования Arduino исходный текст программы принято называть _____ , что можно перевести как «набросок, эскиз».

2.6 _____ – это миниатюрный компьютер, изготовляемый в виде печатной платы малых размеров.

2.7 Главным отличием Raspberry Pi от персонального компьютера является наличие на плате _____ и поддержка популярных интерфейсов для подключения периферийных устройств (UART (Serial); I2C/TWI; SPI).

2.28. На Raspberry Pi может быть установлена операционная система (ОС), поддерживающая ____ – процессоры.

2.9 ____ – это объектно-ориентированный, интерпретируемый, переносимый язык сверхвысокого уровня.

2.10 _____ содержит несколько элементов различных или одинаковых типов и может динамически изменять размер.

3 Вопросы на установление соответствия.

3.1 Установить соответствие компонентов аппаратно-программных платформ с их описанием:

1) Центральный процессор (CPU)	a) Основной вычислительный элемент компьютера, который выполняет инструкции программы.
2) Ввод-вывод (I/O) порты	b) Временное хранилище данных и программ, используемое компьютером при выполнении задач.
3) Оперативная память (RAM)	c) Интерфейсы, которые позволяют компьютеру обмениваться данными с внешними устройствами.
4) Флэш-память (Flash Memory)	d) Постоянное хранилище данных и программ, которое сохраняет информацию после выключения питания.

3.2 Установить соответствие элементов архитектуры Arduino с их функциями:

1) Микроконтроллер ATmega328P	a) Постоянное хранилище данных, которое можно использовать для сохранения информации между запусками программы.
2) Цифровые пины ввода/вывода	b) Пины, предназначенные для аналогового ввода, позволяющие считывать аналоговые значения, например, с датчиков.
3) Аналоговые пины ввода	c) Пины, которые могут работать как цифровые входы или выходы для подключения и управления различными устройствами.
4) EEPROM (Электрически стираемое и программируемое постоянное запоминающее устройство)	d) Постоянное хранилище данных, которое можно использовать для сохранения информации между запусками программы.

3.3 Установить соответствие элементов архитектуры Raspberry Pi с их функциями:

1) Оперативная память (RAM)	a) Временное хранилище данных и программ, используемое Raspberry Pi при выполнении задач.
2) Центральный процессор (CPU)	b) Основной вычислительный элемент Raspberry Pi, который выполняет операционную систему и программы.
3) HDMI порт	c) Пины, которые могут использоваться для подключения и управления внешними устройствами, такими как сенсоры и моторы.
4) GPIO (General Purpose Input/Output) пины	d) Порт, который позволяет подключить Raspberry Pi к монитору или телевизору для вывода видео и аудио сигнала.

3.4 Установить соответствие:

1) Указатель	a) Переменная, которая содержит адрес в памяти другой переменной.
2) Адрес переменной	b) Уникальное значение, которое указывает на местонахождение переменной в памяти.
3) Разыменование указателя	c) Операция, которая позволяет получить значение, на которое указывает указатель.
4) Арифметика указателей	d) Манипуляции с указателями, такие как сложение или вычитание числа, которое влияет на текущий адрес указателя.

3.5 Установить соответствие:

1) Указатель на функцию	a) Тип данных, который может хранить адрес функции и позволяет вызывать функцию, на которую он указывает.
2) Обратный вызов (callback)	b) Механизм, который позволяет передавать функцию как аргумент другой функции и вызывать ее позднее.
3) Функция высшего порядка	c) Функция, которая может принимать другие функции в качестве аргументов или возвращать их как результат.
4) Указатель на член структуры	d) Указатель, который указывает на конкретное поле структуры и позволяет получить доступ к нему.

3.6 Установить соответствие терминов и понятий, связанных с многопоточным программированием в языке C, с их описанием:

1) Поток (thread)	a) Выполняющаяся часть программы, которая может работать параллельно с другими потоками.
2) Синхронизация	b) Координация выполнения потоков, чтобы избежать гонок данных и других проблем многопоточности.
3) Критическая секция	c) Участок кода, который должен выполняться только одним потоком в данный момент времени.
4) Мьютекс (mutex)	d) Объект, используемый для блокировки доступа к ресурсам, позволяя только одному потоку использовать их в определенный момент времени.

3.7 Установить соответствие:

1) Перечисление (enum)	a) Специальный тип данных, который представляет собой набор именованных целочисленных констант.
2) Перечислимый тип (enum type)	b) Абстрактный тип данных, который может содержать набор констант из перечисления.
3) Элемент перечисления	c) Каждый элемент в перечислении, которому присвоено уникальное целочисленное значение.
4) Инициализация перечисления	d) Процесс задания значений для элементов перечисления при его создании.

3.8 Установить соответствие:

1) Ethernet кабель (CAT6)	a) Используется для передачи данных в высокоскоростных сетях, таких как Gigabit Ethernet.
2) Витая пара (Twisted Pair)	b) Используется для соединения компьютеров в локальной сети.
3) Оптоволоконный кабель (Fiber Optic)	c) Обеспечивает высокую пропускную способность и долгие расстояния передачи данных.
4) Коаксиальный кабель (Coaxial)	d) Ранее часто использовался для подключения телевизионных сетей.

3.9 Установить соответствие:

1) Raspbian (теперь известная как Raspberry Pi OS)	a) Оптимизирована для мультимедийного воспроизведения.
2) Ubuntu Mate	b) Основана на Debian Linux.
3) OSMC (Open Source Media Center)	c) Предназначена для создания игровых консолей.
4) RetroPie	d) Имеет графический интерфейс, подходящий для начинающих.

3.10 Установить соответствие:

1) PyCharm	a) Интерактивная среда для выполнения кода Python по строкам, обычно используется для анализа данных.
2) Visual Studio Code	b) Простая среда Python, предустановленная с Python, которую можно использовать для быстрых экспериментов.

3) Jupyter Notebook	c) Интегрированная среда разработки (IDE) для Python с множеством функций и поддержкой сторонних плагинов.
4) IDLE	d) Легковесная среда с расширяемым функционалом и хорошей поддержкой Python.

4 Вопросы на установление последовательности

4.1 Расставьте этапы выполнения программы на микроконтроллере в правильном порядке:

а) Инициализация портов ввода-вывода.

б) Обработка данных и принятие решений.

в) Отправка данных на вывод.

г) Ожидание событий или внешних сигналов.

д) Чтение данных с датчика.

4.2 Установите правильный порядок действий при разработке встроенной системы на микроконтроллере:

а) Определение требований к системе.

б) Проектирование аппаратной части.

в) Тестирование и верификация системы.

г) Массовое производство и внедрение.

д) Написание и отладка программного кода.

4.3 Установите правильный порядок выполнения этапов для взаимодействия с датчиком света и светодиодом на платформе Arduino:

а) Чтение данных с датчика света и сохранение их в переменной.

б) Определение условия (например, если яркость меньше определенного значения).

в) Включение или выключение светодиода в зависимости от условия.

г) Запуск бесконечного цикла в функции void loop() для постоянного мониторинга и реагирования на изменения яркости.

д) Подключение датчика света к платформе Arduino и указание используемого порта.

е) Инициализация датчика света и светодиода в функции void setup().

ж) Включение и настройка датчика света для считывания данных о яркости.

4.4 Расставьте шаги в правильном порядке для начала работы с Raspberry Pi:

а) Вставка microSD-карты с операционной системой в слот на Raspberry Pi.

б) Подключение клавиатуры и мыши к USB-портам Raspberry Pi.

в) Подключение Raspberry Pi к источнику питания (например, микро-USB).

г) Подключение HDMI-кабеля к монитору и Raspberry Pi (если нужен вывод на монитор).

д) Подключение кабеля Ethernet (или настройка Wi-Fi, если необходимо).

е) Включение Raspberry Pi, либо путем включения питания, либо нажатием на кнопку питания (в зависимости от модели).

4.5 Установите правильный порядок для управления Raspberry Pi удаленно через SSH:

а) Получение IP-адреса Raspberry Pi в локальной сети.

б) Открытие терминала на компьютере и ввод команды SSH с указанием IP-адреса Raspberry Pi.

в) Ввод пароля или ключа SSH для аутентификации.

г) Успешное подключение к Raspberry Pi через SSH и возможность управления им удаленно.

д) Включение SSH на Raspberry Pi (если не активировано).

4.6 Расставьте шаги в правильном порядке для установки операционной системы на Raspberry Pi с использованием карты microSD:

а) Выбор образа операционной системы и целевого устройства (microSD-карты).

б) Начало процесса записи образа на microSD-карту.

в) Загрузка операционной системы Raspberry Pi на компьютер.

г) Запуск программы для записи образа операционной системы на microSD-карту (например, Etcher).

д) Извлечение microSD-карты из компьютера и вставка её в Raspberry Pi.

е) Подключение Raspberry Pi к источнику питания и включение его.

4.7 Расставьте шаги в правильном порядке для настройки доступа к веб-интерфейсу (веб-панели) Raspberry Pi:

а) Настройка веб-сервера для прослушивания на нужном порту и директории.

б) Публикация веб-страницы или приложения в директории веб-сервера.

в) Установка веб-сервера (например, Apache или Nginx) на Raspberry Pi.

г) Написание веб-страницы или приложения, которые будут отображаться через веб-интерфейс.

д) Открытие браузера на компьютере и ввод IP-адреса Raspberry Pi для доступа к веб-интерфейсу.

4.8 Установите последовательность разработки аппаратной части в инфокоммуникационных системах:

а) Проектирование схемы.

б) Печатная плата и монтаж.

в) Использование микроконтроллеров.

г) Сборка корпуса.

4.9 Расставьте этапы разработки программы для Arduino в правильном порядке:

а) Подключение к плате Arduino.

б) Загрузка программы на плату.

в) Написание кода в среде Arduino IDE.

г) Загрузка библиотек, если необходимо.

4.10 Расставьте этапы настройки Raspberry Pi в правильном порядке:

а) Подключение к Wi-Fi сети

б) Загрузка операционной системы на SD-карту

в) Подключение к монитору и клавиатуре

г) Установка необходимых программ и библиотек

Шкала оценивания: 4 балльная.

Критерии оценивания:

Каждый вопрос (задание) в тестовой форме оценивается по дихотомической шкале: выполнено – 0,5 балла, не выполнено – 0 баллов.

Применяется следующая шкала перевода баллов в оценку по 5-балльной шкале:

- **4 балла** соответствуют оценке **«отлично»**;
- **3 балла** – оценке **«хорошо»;**
- **2 балла** – оценке **«удовлетворительно»;**
- **1 балл и менее** – оценке «**неудовлетворительно».**

1.3 ТЕМЫ РЕФЕРАТОВ

Раздел 1. Язык С

1. История и эволюция языка С.
2. Основные концепции и структуры данных в языке С.
3. Функции и модульность в языке С.
4. Управление памятью в С: указатели и динамическое выделение памяти.
5. Обработка ошибок и отладка программ на С.
6. Создание и использование библиотек в С.
7. Многопоточное программирование в С.
8. Сетевое программирование на языке С.
9. Оптимизация кода на С для повышения производительности.
10. Безопасность программирования на С и уязвимости.

Раздел 2. Аппаратно-программные платформы инфокоммуникационных систем

1. Использование Raspberry Pi в инфокоммуникационных системах.
2. Программирование и настройка Raspberry Pi для проектов связи и автоматизации.
3. Аппаратно-программные платформы для Интернета вещей (IoT).
4. Использование Arduino в инфокоммуникационных системах.
5. Проекты на Arduino для управления и мониторинга инфокоммуникационных сетей.
6. Аппаратно-программные платформы для обработки и анализа данных в инфокоммуникационных системах.
7. Сетевые протоколы и коммуникации в инфокоммуникационных системах.
8. Безопасность и защита данных на аппаратно-программных платформах инфокоммуникаций.
9. Аппаратно-программные платформы для облачных инфраструктур.
10. Тренды и будущее аппаратно-программных платформ в инфокоммуникационных системах.

Шкала оценивания: 6 балльная.

Критерии оценивания :

5-6 баллов (или оценка **«отлично»**) выставляется обучающемуся, если тема реферата раскрыта полно и глубоко, при этом убедительно и аргументированно изложена собственная позиция автора по рассматриваемому вопросу; структура реферата логична; изучено большое количество актуальных источников, грамотно сделаны ссылки на источники; самостоятельно подобран яркий иллюстративный материал; сделан обоснованный убедительный вывод; отсутствуют замечания по оформлению реферата.

3-4 баллов (или оценка **«хорошо»**) выставляется обучающемуся, если тема реферата раскрыта полно и глубоко, сделана попытка самостоятельного осмысления темы; структура реферата логична; изучено достаточное количество источников, имеются ссылки на источники; приведены уместные примеры; сделан обоснованный вывод; имеют место незначительные недочеты в содержании и (или) оформлении реферата.

1-2 баллов (или оценка **«удовлетворительно»**) выставляется обучающемуся, если тема реферата раскрыта неполно и (или) в изложении темы имеются недочеты и ошибки; структура реферата логична; количество изученных источников менее рекомендуемого, сделаны ссылки на источники; приведены общие примеры; вывод сделан, но имеет признаки неполноты и неточности; имеются замечания к содержанию и (или) оформлению реферата.

0 баллов (или оценка **«неудовлетворительно»**) выставляется обучающемуся, если содержание реферата имеет явные признаки плагиата и (или) тема реферата не раскрыта и (или) в изложении темы имеются грубые ошибки; материал не структурирован, излагается

непоследовательно и сбивчиво; количество изученных источников значительно менее рекомендуемого, неправильно сделаны ссылки на источники или они отсутствуют; не приведены примеры или приведены неверные примеры; отсутствует вывод или вывод расплывчат и неконкретен; оформление реферата не соответствует требованиям.

2 ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ

2.2 БАНК ВОПРОСОВ И ЗАДАНИЙ В ТЕСТОВОЙ ФОРМЕ

1 Вопрос в закрытой форме.

1.1. В языке программирования Си в записи выражения int prop[3][4][6];
а) Ошибка вследствие применения ключевого слова
б) Нет ошибок
в) Ошибка вследствие применения служебных символов

1.2. В языке программирования Си в записи выражения int i, *ip, (*ipp)[6]:
а) Ошибка вследствие применения ключевого слова
б) Нет ошибок
в) Ошибка вследствие применения служебных символов

1.3. В языке программирования Си в записи a=123; присутствует:
а) Логическая операция
б) Операция присваивания
в) Арифметическая операция

1.4. В языке программирования Си в записи a+=123; присутствует:
а) Арифметическая операция
б) Запись ошибочна
в) Логическая операция

1.5. В языке программирования Си в записи a/=123; присутствует:
а) Логическая операция
б) Арифметическая операция
в) Запись ошибочна

1.6. В языке программирования Си в записи a-=123; присутствует:
а) Логическая операция
б) Арифметическая операция
в) Запись ошибочна

1.7. В языке программирования Си в записи a&=123; присутствует:
а) Логическая операция И
б) Побитовая операция И
в) Запись ошибочна

1.8. В языке программирования Си в записи a=b & 0xF; присутствует:
а) Запись ошибочна
б) Побитовая операция И
в) Логическая операция

1.9. В языке программирования Си в записи (a==b && b=3) присутствует:
а) Запись ошибочна
б) Логическая операция И
в) Побитовая операция И

1.10. В языке программирования Си выражения инкремента a++ и ++a:
а) Всегда эквивалентны при использовании
б) Не всегда эквивалентны при использовании
в) Применимы только в языке Си++

1.11.
```
int   i, j, k;
i=j+34;
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Ошибка во второй строке
б) Нет синтаксических ошибок
в) Ошибка в третьей строке

1.12.
```
int   i, j, k;
i=j+34
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка во второй строке
в) Ошибка в третьей строке

1.13.
```
integer i, j, k;
i=j+34;
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в первой строке
в) Ошибка в третьей строке

1.14.
```
double i, j, k;
i=j+.34;
k+= i;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка во второй строке

1.15.
```
double int, jnt, knt;
knt=jnt+.34;
knt/= 2.;
```
В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в первой строке
в) Ошибки во второй и третьей строках

1.16.

```
int   i; j; k;
i=j+34;
k+= i;
```

В приведенном фрагменте программы (рисунок):
а) Нет синтаксических ошибок
б) Ошибка в первой строке
в) Ошибки во второй и третьей строках

1.17.

```
float f1;
int j,k;
j=(int)f1;
```

В приведенном фрагменте программы (рисунок):
а) Ошибка в первой строке
б) Нет синтаксических ошибок
в) Ошибки во второй и третьей строках

1.18.

```
float f1;
int j;k;
j=(int)f1;
```

В приведенном фрагменте программы (рисунок):
а) Ошибки в третьей строке
б) Ошибка во второй строке
в) Нет синтаксических ошибок

1.19.

```
float f1;
int j,k
j=(int)f1;
```

В приведенном фрагменте программы (рисунок):
а) Ошибки во второй и третьей строках
б) Ошибка во второй строке
в) Нет синтаксических ошибок

1.20.

```
float f1;
int j,k;
if(j==k);;
```

В приведенном фрагменте программы (рисунок):
а) Ошибка в первой строке
б) Нет синтаксических ошибок
в) Ошибки в третьей строке

1.21.

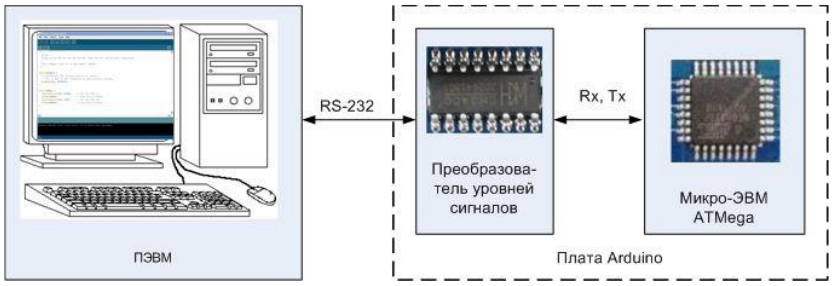


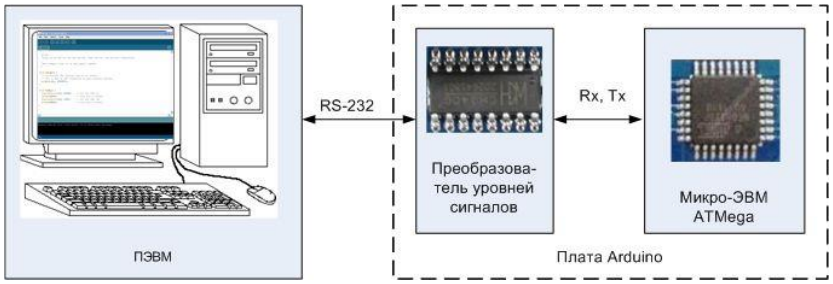
Рисунок иллюстрирует:

а) соединение для загрузки программного кода в платформу Arduino

б) все перечисленные варианты правильные

в) соединение для обмена данными с ПЭВМ в процессе работы системы на базе Arduino

1.22.



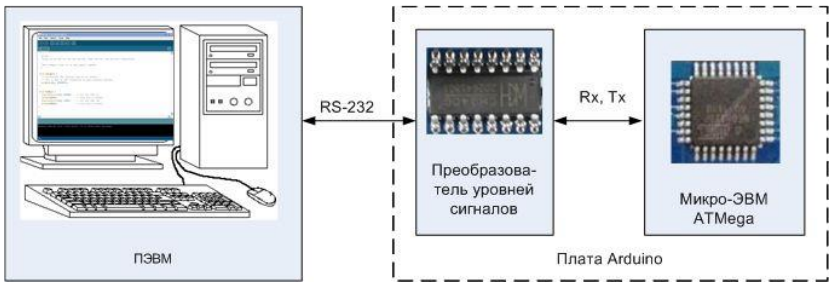
На рисунке показано:

а) загрузка программного кода из ПЭВМ по интерфейсу USB

б) загрузка программного кода из ПЭВМ по интерфейсу RS-232

в) обмен данными между ПЭВМ и Arduino по интерфейсу USB

1.23.



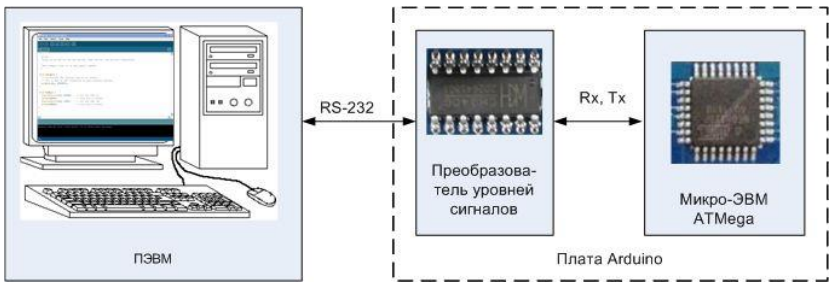
Преобразователь уровней сигналов на рисунке:

а) преобразует как уровни сигналов, так и поток данных из ПЭВМ

б) преобразует только уровни сигналов для согласования

в) устанавливается изготовителем на элитных платах исключительно с коммерческой целью

1.24.



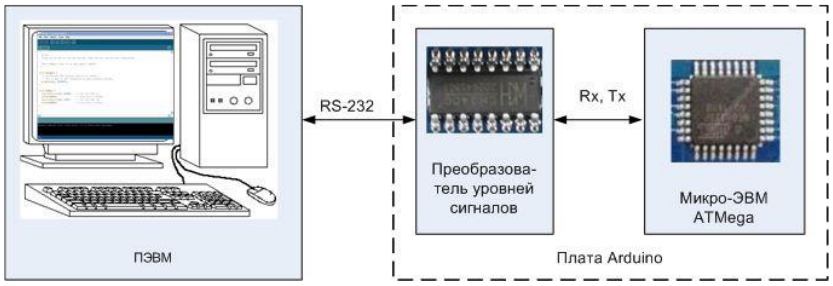
Среда разработки для платформы Arduino (рисунок):

а) включает специальное программное обеспечение для ПЭВМ, коммутационные средства для соединения и не требует предустановки содержимого памяти команд микро-ЭВМ

б) включает специальное программное обеспечение для ПЭВМ, предустановленное содержимое памяти команд микро-ЭВМ и коммутационные средства для соединения

в) включает предустановленное содержимое памяти команд микро-ЭВМ, коммутационные средства для соединения и использует стандартные средства ОС Windows

1.25.



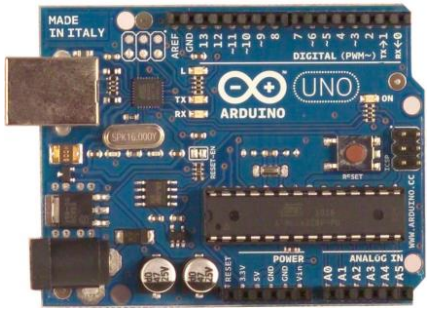
Специальное программное обеспечение на ПЭВМ для среды Arduino выполняет функции:

а) только для трансляции пользовательских программ в машинный код и передачи их на плату Arduino

б) трансляции пользовательских программ в машинный код, передачи их на плату Arduino, запуска записанной программы и при необходимости - обмена данными с платформой Arduino

в) трансляции пользовательских программ в машинный код, передачи их на плату Arduino, запуска записанной программы и её динамической отладки в процессе исполнения

1.26.



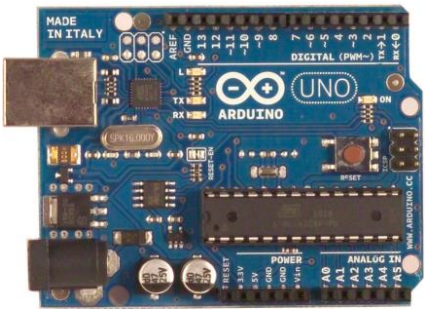
Среда разработки программ Arduino основана:

а) на языке Pascal

б) на языке C

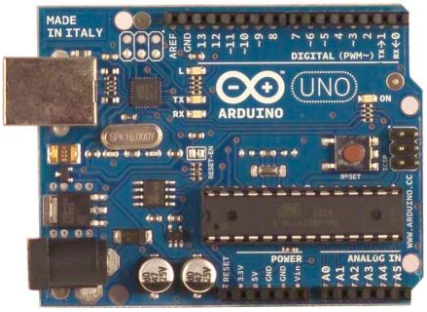
в) на собственном языке Arduino

1.27.



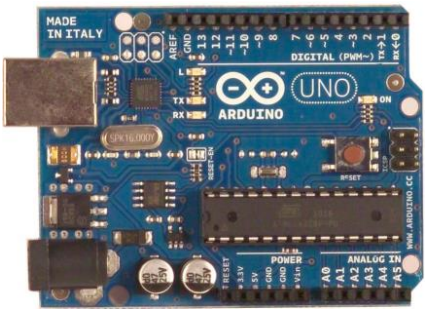
Структурно программа (sketch) для среды Arduino содержит:
а) функцию main() и пользовательские функции
б) функции setup(), loop(), пользовательские функции
в) только пользовательские функции

1.28.



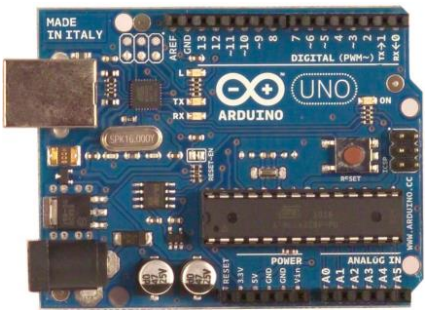
Переменные в программе (sketch) могут быть определены:
а) только внутри функций setup() и loop()
б) в любом структурном подмодуле программы
в) только в начале программы, до функции setup()

1.29.



Функция setup() в программе (sketch) Arduino:
а) вызывается при необходимости из функции loop()
б) выполняется первой, до вызова остальных функций
в) порядок выполнения функций задается программистом

1.30.



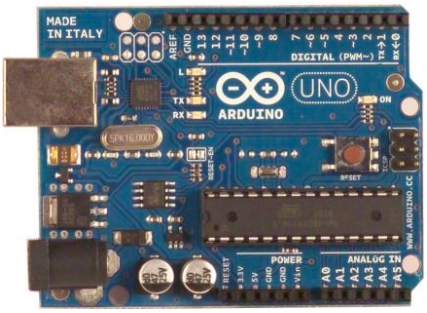
Функция loop в программе (sketch):

а) вызывается при необходимости из функции setup()

б) образует бесконечный цикл

в) вызывается однократно

1.31.



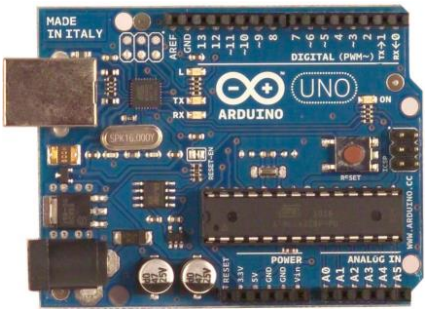
В программе (sketch) Arduino:

а) пользовательские функции могут быть определены только внутри структурной единицы loop()

б) имеется возможность создавать и использовать пользовательские функции

в) допускается использовать только библиотечные функции

1.32.



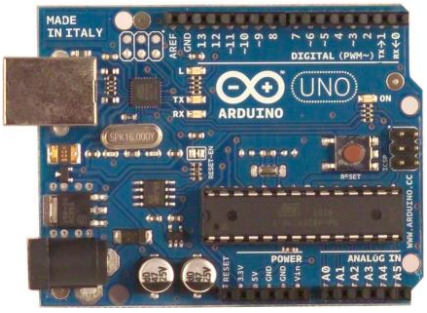
Основное назначение функции setup() в программе (sketch) Arduino:

а) объявление и задание начальных значений всех переменных

б) обеспечение необходимых настроек оборудования на нужные режимы работы

в) запуск механизма загрузки программного кода в память команд микро-ЭВМ платы Arduino

1.33.



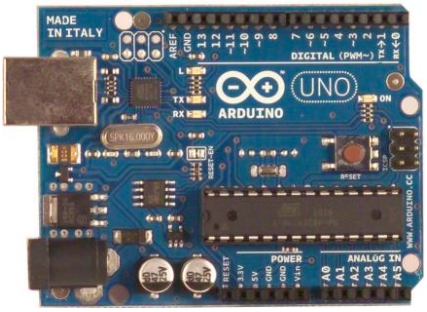
Основное назначение функции loop() в программе (sketch) Arduino:

а) эта функция однократно запускает заданный программистом алгоритм

б) организация бесконечного цикла для повторения заданного программистом алгоритма

в) эта функция предназначена для осуществления отладки программы и позволяет прервать процесс исполнения при возникновении определенных условий

1.34.



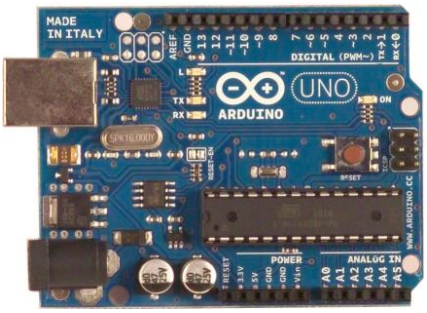
Программа (sketch) Arduino по эффективности (быстродействию и объему кода):

а) существенно лучше программных продуктов, написанным на языке Си и ассемблер, так как оптимизирована именно для Arduino

б) потенциально уступает продуктам, написанным на языке Си и ассемблер

в) все перечисленные варианты равнозначны

1.35.



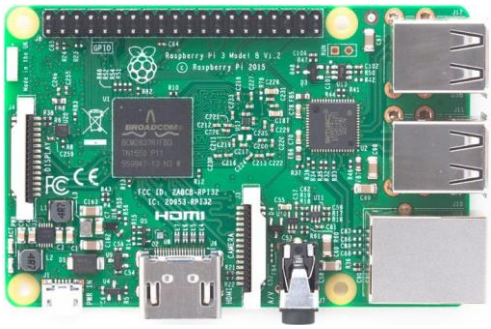
Программа (sketch) Arduino при формировании исполнимого (машинного) кода:

а) это интерпретируемый текст, наподобие Basic или Javaб) образует бесконечный цикл

б) преобразуется в текст на языке Си, а затем компилируется

в) сразу компилируется в машинные коды специальным компилятором с языка Arduino

1.36.



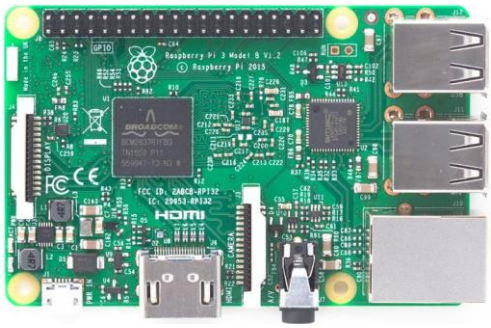
Платформа Raspberry Pi представляет собой:

а) Arduino-подобную платформу, предназначенную для управления объектами, но имеющую более высокую производительность

б) миниатюрный компьютер, к которому можно подключить стандартные монитор, клавиатуру и мышь

в) приставку к персональному компьютеру, предназначенную для расширенной передачи управляющих сигналов управляемым объектам

1.37.



Программное обеспечение платформы Raspberry Pi базируется на операционной системе:

а) как и в Arduino, применение операционной системы не предусмотрено

б) основанной на ОС Linux

в) Microsoft Windows XP или 7

1.38.

Обладает ли платформа Raspberry Pi (до модели Pi 3 включительно) встроенными средствами для управления электронными объектами?

а) да, включая цифровые линии ввода-вывода, интерфейсы I2C, SPI, RS-232, а также линии аналогового ввода-вывода

б) да, включая цифровые линии ввода-вывода и интерфейсы I2C, SPI, RS-232

в) нет, как и персональный компьютер. Но можно использовать оборудование, подключаемое посредством USB

1.39.

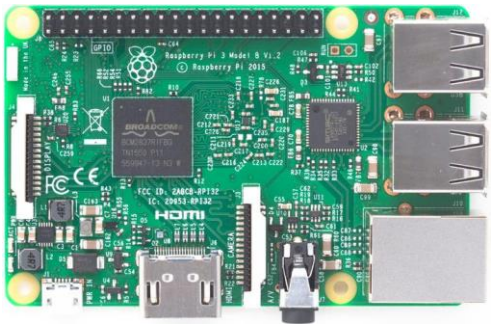
Основной сферой использования Raspberry Pi следует считать:

а) замену платформ Arduino в тех разработках, где требуется большая вычислительная производительность или функциональность

б) всё перечисленное

в) применение в разнообразных разработках электронных изделий энтузиастами, не являющимися профессиональными программистами

1.40.



Для разработки программ на платформе Raspberry Pi:

а) как и для Arduino, требуется дополнительный персональный компьютер, с которого и будет происходить загрузка программы для исполнения

б) достаточно возможностей самой платформы

в) средства разработки ограничены, используют только набор готовых программ, предоставляемый разработчиком платформы

1.41.

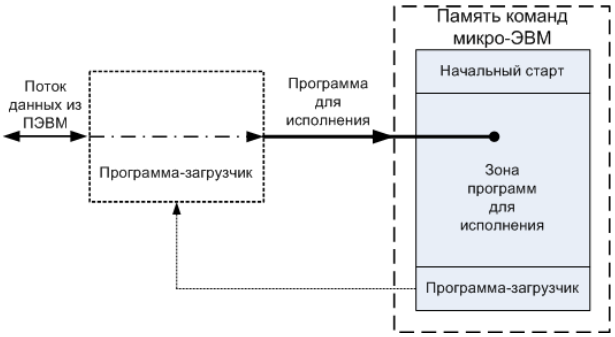


Рисунок иллюстрирует:

а) процесс обмена данными между ПЭВМ и микро-ЭВМ

б) процесс занесения программного кода в микро-ЭВМ

в) определение степени загрузки процессора микро-ЭВМ

1.42.

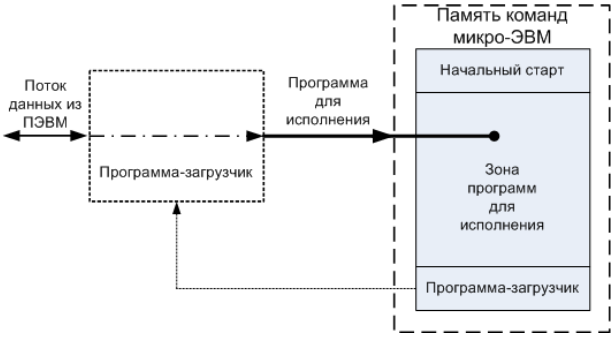


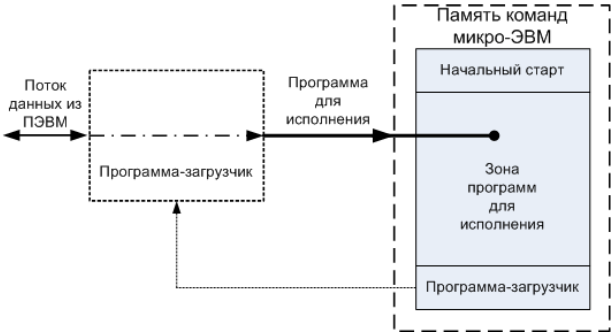
Схема загрузки программного кода на рисунке:

а) пригодна для микро-ЭВМ при наличии команд «самозагрузки»

б) пригодна только для платформ Arduino

в) универсальна для любой микро-ЭВМ

1.43.



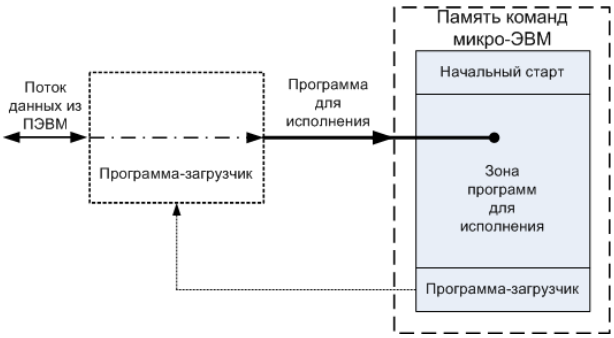
На рисунке показано:

а) загрузка данных из ПЭВМ в память платформы Arduino по интерфейсу USB

б) загрузка программного кода из ПЭВМ по интерфейсу RS-232

в) загрузка программного кода из ПЭВМ по интерфейсу SPI

1.44.



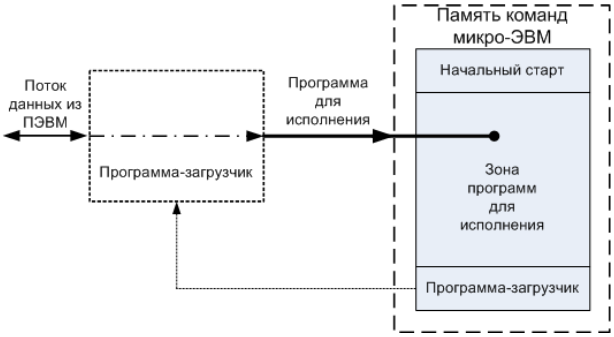
На рисунке для памяти команд микро-ЭВМ:

а) сверху - вниз показана последовательность исполнения функциональных частей

б) показано распределение памяти Arduino по признаку выполняемых функций

в) выделены зоны исполняемого кода и неисполняемых данных

1.45.



На рисунке для компонент памяти команд микро-ЭВМ обязательная последовательность

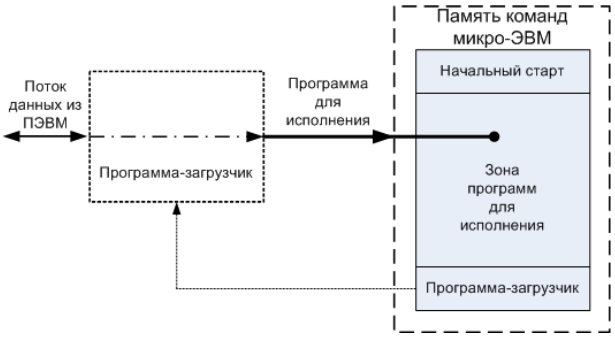
исполнения:

а) 1) Начальный старт 2) в зависимости от дополнительных условий - загрузчик или программа для исполнения

б) 1) Начальный старт - 2) Программа-загрузчик 3) Загруженная программа

в) 1) Начальный старт - 2) Загруженная программа 3) Программа-загрузчик

1.46.



Зона программ для исполнения содержит:

а) загруженный с ПЭВМ текст программы на языке Си

б) загруженный с ПЭВМ двоичный код

в) предустановленное изготовителем микро-ЭВМ фиксированное содержимое

1.47.

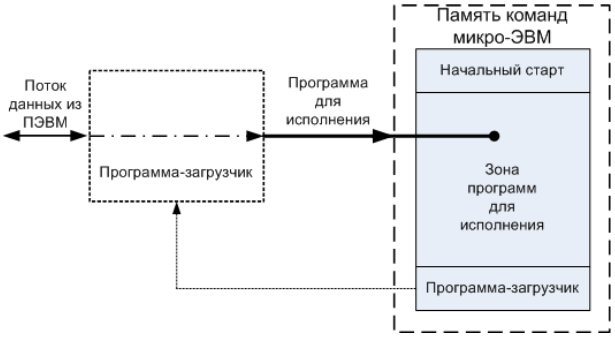
Программа-загрузчик:

а) всегда выполняется после старта

б) выполняется в зависимости от дополнительных условий

в) содержит свой дубликат в памяти данных в процессе загрузки

1.48.



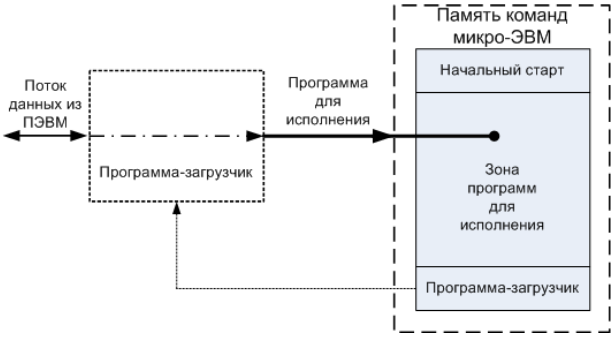
В процессе нормальной загрузки (см. рисунок):

а) полностью переписывается содержимое всей памяти после предварительного копирования загрузчика в память данных

б) возможно изменение любой части зоны программ для исполнения

в) полностью переписывается содержимое всей памяти, кроме зоны «начальный старт»

1.49.



Загрузка программ для платформы Arduino доступна и происходит в соответствии с рисунком:

а) всегда, для аппаратно-исправной платформы

б) только при наличии предустановленного программного кода

в) невозможна ни при каких условиях, если нет предустановленного кода и микро-

ЭВМ не установлена в панельке

1.50.

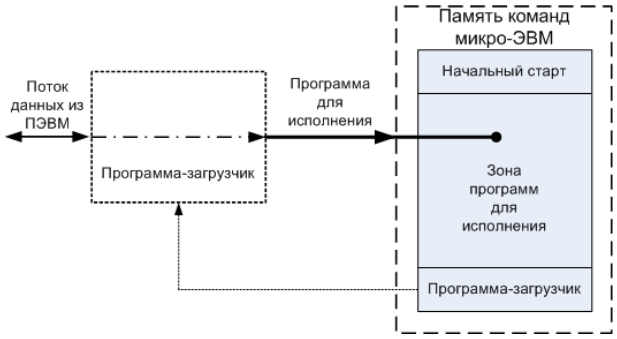


Рисунок иллюстрирует процесс загрузки программного кода для платформы Arduino:

а) только для снабженной интерфейсным разъемом USB
б) для любой из перечисленных
в) только для снабженной интерфейсным разъемом RS-232

1.51.

```
1
for(i=0;i<4;i++)
{
 a=mass[i];
};
```

В тексте фрагмента программы приведен:
а) Оператор цикла с пост-условием выхода
б) Оператор цикла для известного количества итераций
в) Оператор цикла с пред-условием выхода

1.52.

```
2
do
{
 a=mass[i];
}
while(i<0x12);
```

В тексте фрагмента программы приведен
а) Оператор цикла для известного количества итераций
б) Оператор цикла с пост-условием выхода
в) Оператор цикла с пред-условием выхода

1.53.

```
3
while(i<0x12);
{
 a=mass[i];
}
```

В тексте фрагмента программы приведен
а) Оператор цикла для известного количества итераций
б) Оператор цикла с пред-условием выхода
в) Оператор цикла с пост-условием выхода

1.54.

```
switch(i)
{
 case 1:
   a=1; break;
 case 2:
   a=10; break;
 default:
   a=0; break;
}
```

В тексте фрагмента программы приведен
а) Оператор-переключатель и составной оператор
б) Только оператор-переключатель
в) Оператор цикла с пред-условием выхода

1.55.

```
if(a<5)
{
  x=1; y=2;
}
```

В тексте фрагмента программы приведен
а) Только условный оператор
б) Условный оператор и составной оператор
в) Только составной оператор

1.56.

```
if(a<5)x=1;
else y=2;
```

В тексте фрагмента программы приведен
а) Условный оператор и составной оператор
б) Только условный оператор
в) Только составной оператор

1.57.

```
{
 a=1;
 b=2;
 c=3;
}
```

В тексте фрагмента программы приведен
а) Только условный оператор
б) Составной оператор и операторы присваивания
в) Составной оператор и арифметические операторы

1.58.

8

```
for(i=0;;i++)
{
 m[i]=0;
 if(mass[i]==0) break;
}
```

Оператор break (рисунок)

а) Прерывает не только цикл, но и функцию, его содержащую

б) Прерывает выполнение цикла

в) Инициирует продолжение цикла

1.59.

9

```
for(i=0;;)
{
 m[i]=0;
 if(mass[i++]!=0) continue;
}
```

Оператор continue (рисунок)

а) Инициирует повторное выполнение функции, содержащей цикл

б) Инициирует продолжение цикла

в) Прерывает выполнение цикла

1.60.

```
for(i=0;;)
{
 m[i]=0;
 if(mass[i++]!=0) return 1;
}
```

Оператор return (рисунок)

а) Прерывает выполнение цикла

б) Прерывает не только цикл, но и функцию, его содержащую

в) Инициирует повторное выполнение функции, содержащей цикл

1.61.

11

```
structure kalendar
{
 int god;
 int mesjats;
 int denb;
};
```

В тексте фрагмента программы (рисунок) приведено

а) Структура

б) Функция

в) Объединение

1.62.

```
12
 structure menu
 {
  char pervoe_bludo[20];
  char vtoroe_bludo[20];
  char tretje_bludo[20];
 };
```

В тексте фрагмента программы (рисунок) приведено
а) Функция
б) Запись
в) Смесь

1.63.

```
 union mesjats
 {
  int nomer;
  char nazv[20];
 };
```

В тексте фрагмента программы (рисунок) приведено
а) Структура
б) Объединение
в) Функция

1.64.

```
14
 union den_nedeli
 {
  int nomer;
  char nazv[20];
 };
```

В тексте фрагмента программы (рисунок) приведено
а) Смесь
б) Функция
в) Запись

1.65.

```
15
 int fun_sravn(int a, int b)
 {
  if (a==b)return 0;
  else return 1;
 }
```

В тексте фрагмента программы (рисунок) приведено
а) Объединение
б) Функция
в) Структура

1.66.

```
char *fun_sravn(int a, int b)
{
 if (a==b)return "ravno";
 else return "ne ravno";
}
```

В тексте фрагмента программы (рисунок) приведено
а) Объединение
б) Функция
в) Структура

1.67.

```
17
 void kub(int a, int *b)
 {
  *b=a*a*a;
 }
```

В тексте фрагмента программы (рисунок) приведено
а) Объединение
б) Функция
в) Структура

1.68.

```
18
int glob_var1;

void kub(int a)
{
 glob_var1=a*a*a;
}
```

В тексте фрагмента программы (рисунок) приведено
а) Объединение
б) Функция
в) Структура

1.69.

```
#define N 200
#include <stdio.h>
#include "math.h"

main()
{
 int a,b;

 nachalo(a,b);
 obrabotka(N);
}
```

В тексте фрагмента программы (рисунок) конструкция языка #define
а) Определяет число запусков программы

б) Определяет символическое имя с присвоением ему константного значения
в) Осуществляет обращение к процедуре с именем N и параметром 200

1.70.

20

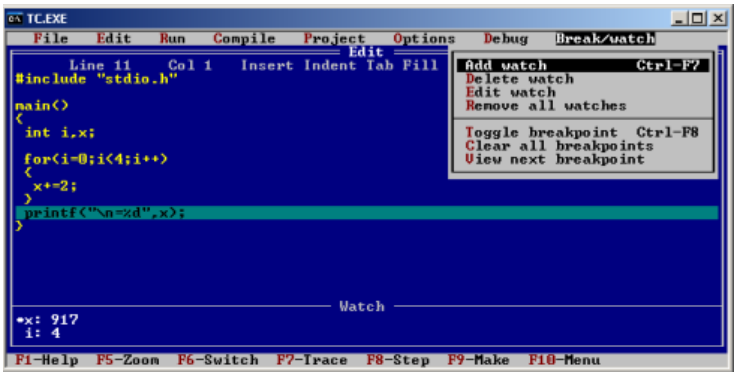
```
#define N 200
#include <stdio.h>
#include "math.h"

main()
{
 int a,b;

 nachalo(a,b);
 obrabotka(N);
}
```

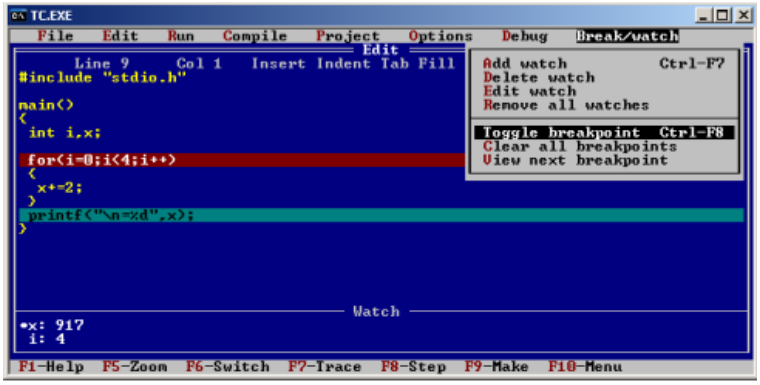
В тексте фрагмента программы (рисунок) конструкция языка #include
а) Вызывает исполнение программы с указанным именем
б) Подключает библиотечные функции
в) Задает параметры среды исполнения функции main()

1.71.



Что показывают записи внизу экрана под словом Watch
а) Количество точек наблюдения 4 и число прогонов программ 917
б) Значения наблюдаемых переменных x=917, i=4 на момент останова программы
в) Количество точек останова (breakpoints) в программе 4, значение исследуемой переменной
(evaluate) x=917

1.72.



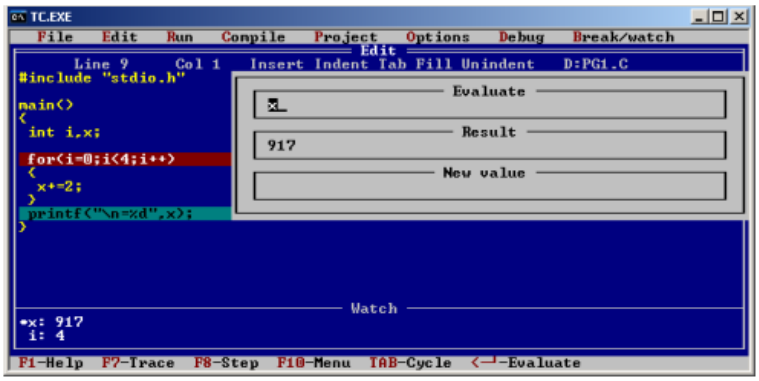
Какая функция отладки осуществляется на экране
а) Переключение точки останова (breakpoint) в противоположное состояние

активности

б) Сброс активности точки останова (breakpoint)

в) Установка точки останова (breakpoint) в состояние активности

1.73.



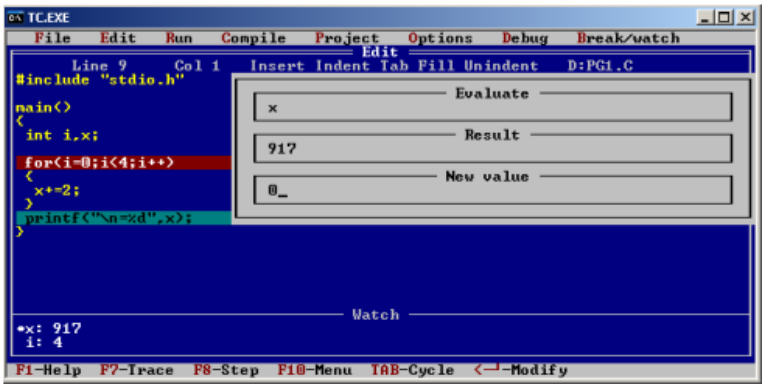
Какая функция отладки осуществляется на экране

а) Наблюдение текущего значения переменной x с помощью отладочного средства evaluate

б) Настройка отладочного средства watch для отображения значения переменных

в) Изменение текущего значения переменной x с помощью отладочного средства evaluate

1.74.



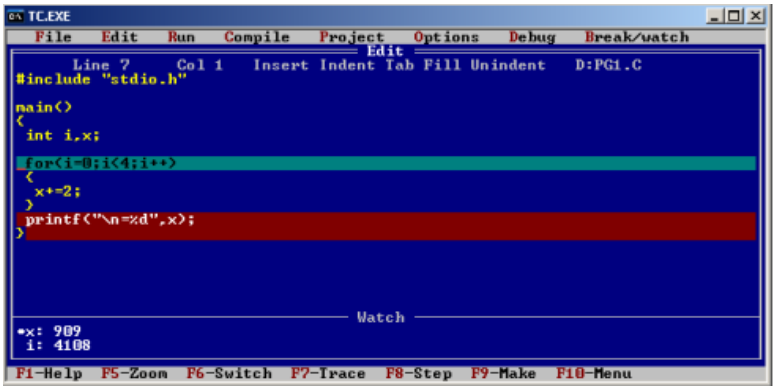
Какая функция отладки осуществляется на экране

а) Изменение текущего значения переменной x с помощью отладочного средства evaluate

б) Настройка отладочного средства watch для отображения значения переменных

в) Наблюдение текущего значения переменной x с помощью отладочного средства watch

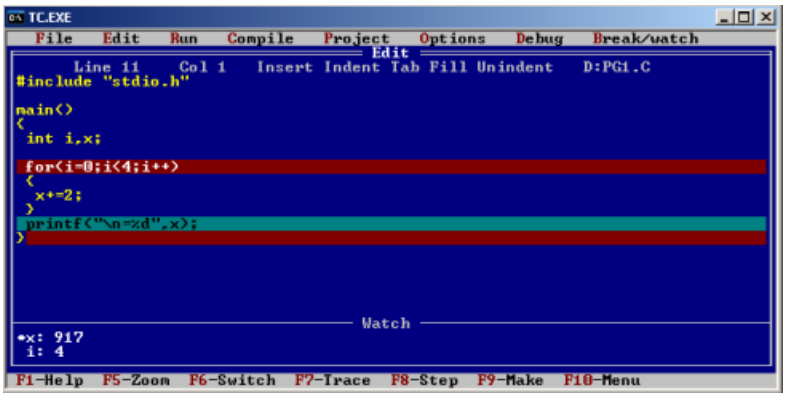
1.75.



Какая функция отладки осуществляется на экране

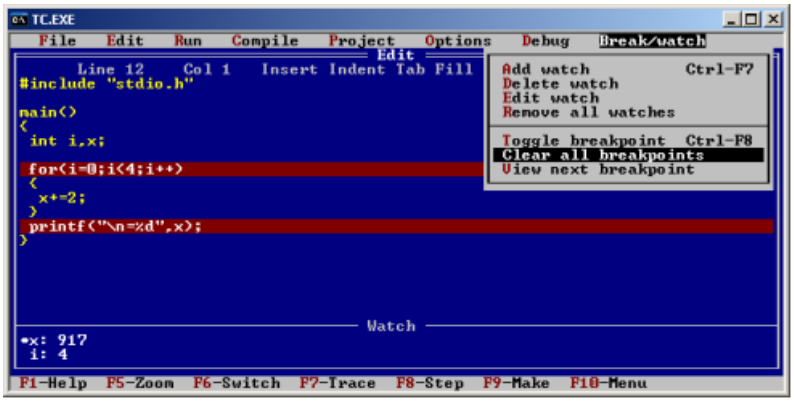
а) Останов в первой точке останова и наблюдение значений переменных
б) Процесс выполнения программы и наблюдение текущих значений переменных
в) Останов во второй точке останова и наблюдение значений переменных

1.76.



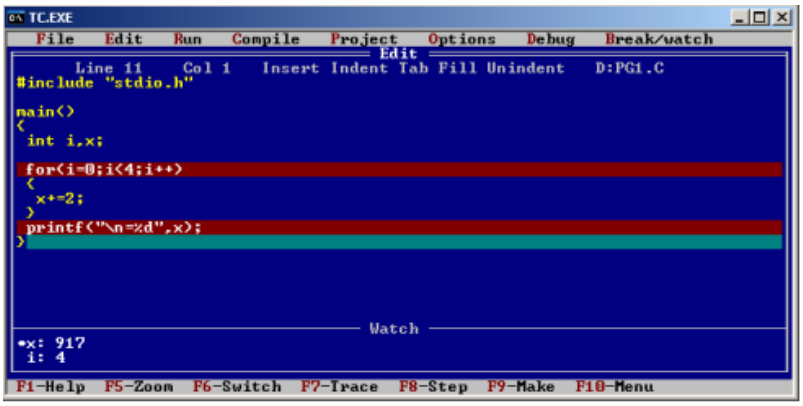
Какая функция отладки осуществляется на экране
а) Процесс выполнения программы и наблюдение текущих значений переменных
б) Останов во второй точке останова и наблюдение значений переменных
в) Останов в первой точке останова и наблюдение значений переменных

1.77.



Какая функция отладки осуществляется на экране
а) Процесс выполнения программы и наблюдение текущих значений переменных
б) Сброс всех точек останова в программе
в) Останов в первой точке останова и наблюдение значений переменных

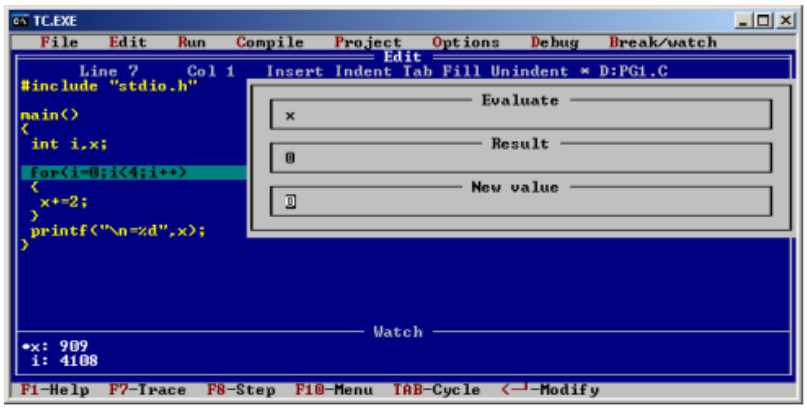
1.78.



Значения переменных после выполнения программы таковы, что анализируя текст программы:
а) Можно было предсказать эти значения до запуска
б) Невозможно было предсказать эти значения до запуска

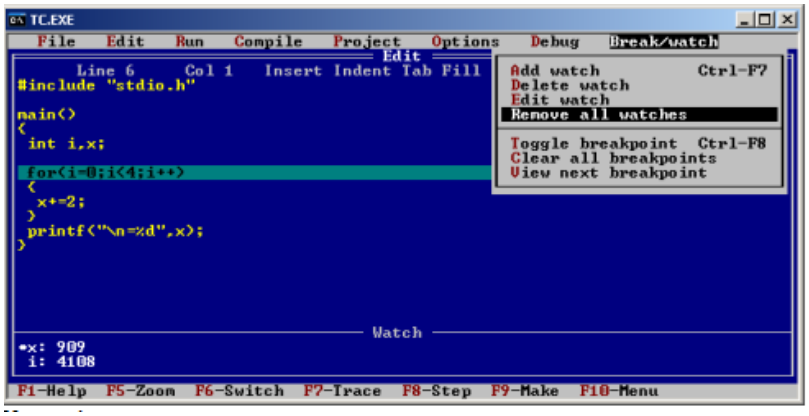
в) Эти значения переменных не связаны с текущим прогоном программы

1.79.



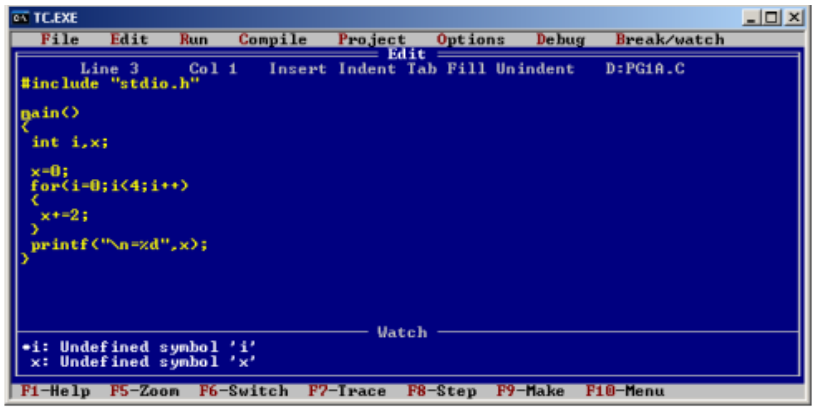
Какая функция отладки осуществляется на экране
а) Останов и наблюдение значений переменных i, x
б) Останов в точке останова и изменение значения переменной x
в) Процесс выполнения программы и наблюдение текущих значений переменных

1.80.



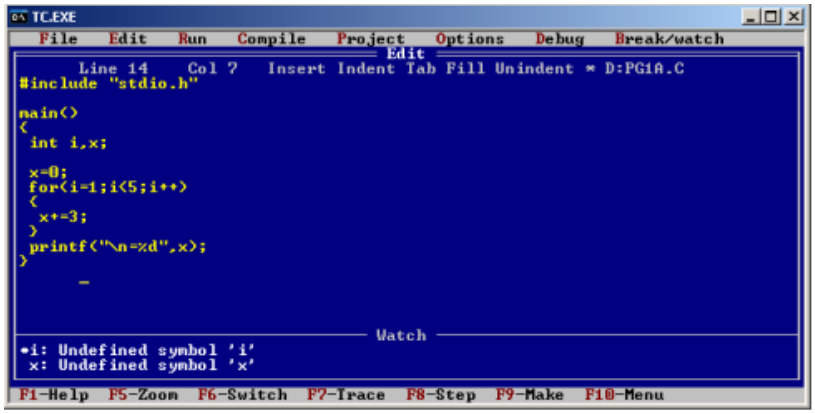
Какая функция отладки осуществляется на экране
а) Сброс значений переменных для точек наблюдения (watches)
б) Удаление всех точек наблюдения (watches)
в) Процесс выполнения программы и наблюдение текущих значений переменных

1.81.



Что отобразится на экране по завершении программы
а) =8
б) 9
в) 8

1.82.



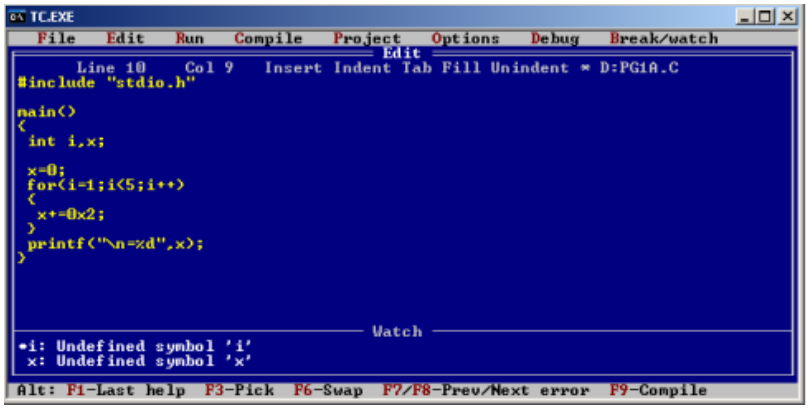
Что отобразится на экране по завершении программы

а) =12

б) 12

в) =9

1.83.



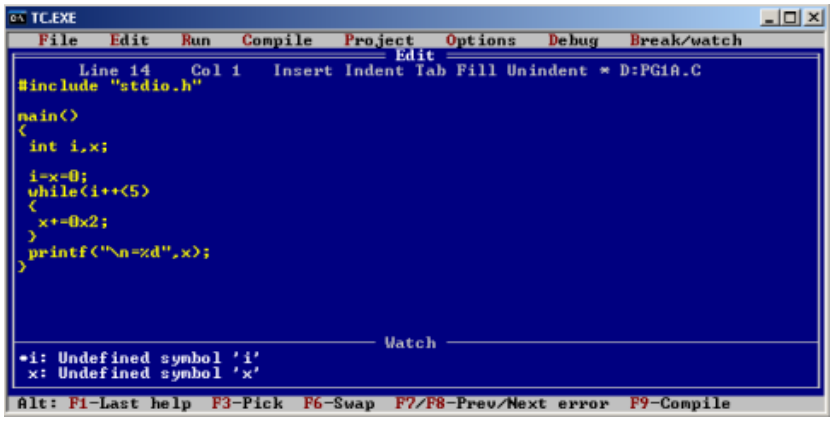
Что отобразится на экране по завершении программы

а) =9

б) =8

в) 12

1.84.



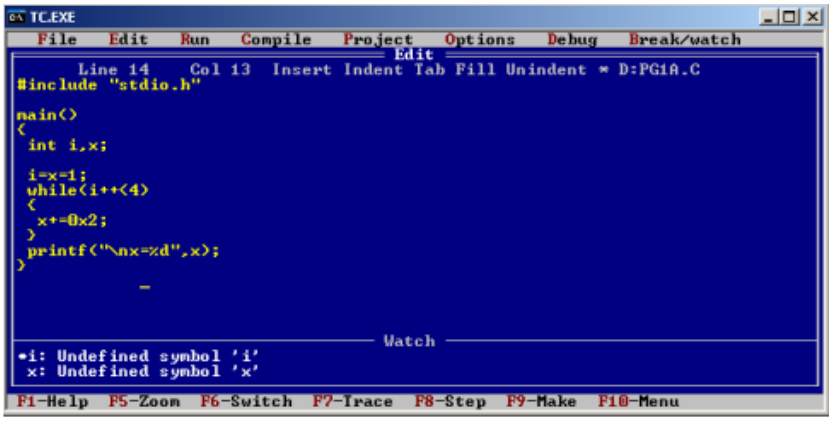
Что отобразится на экране по завершении программы

а) 9

б) =10

в) 10

1.85.



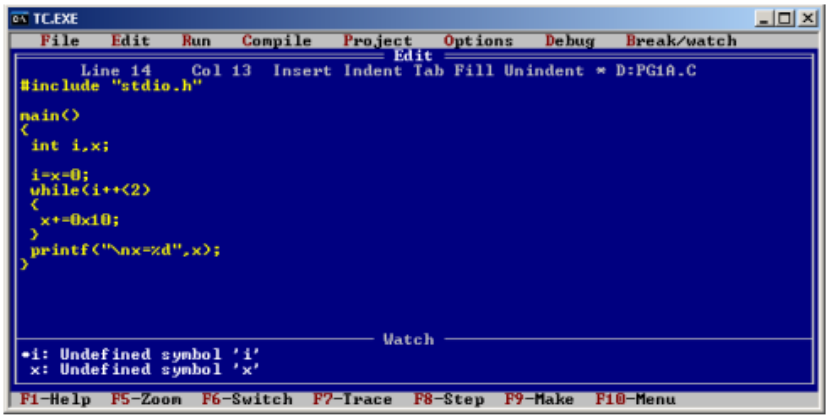
Что отобразится на экране по завершении программы

а) 7

б) x=7

в) =8

1.86.



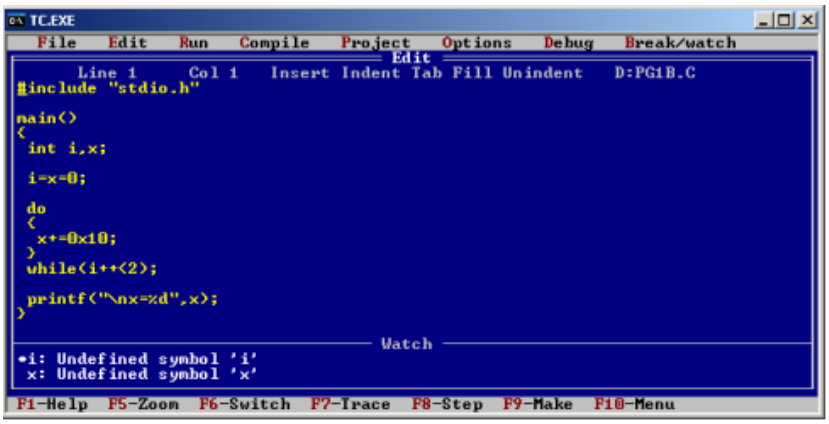
Что отобразится на экране по завершении программы

а) 20

б) x=32

в) 32

1.87.



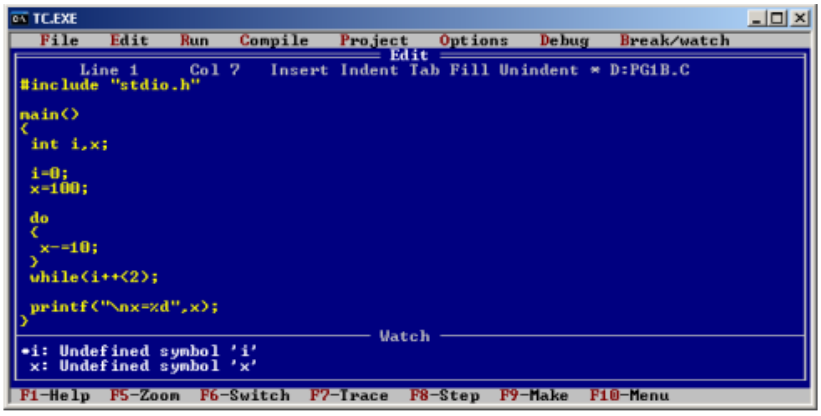
Что отобразится на экране по завершении программы

а) X=48

б) 34

в) X=40

1.88.



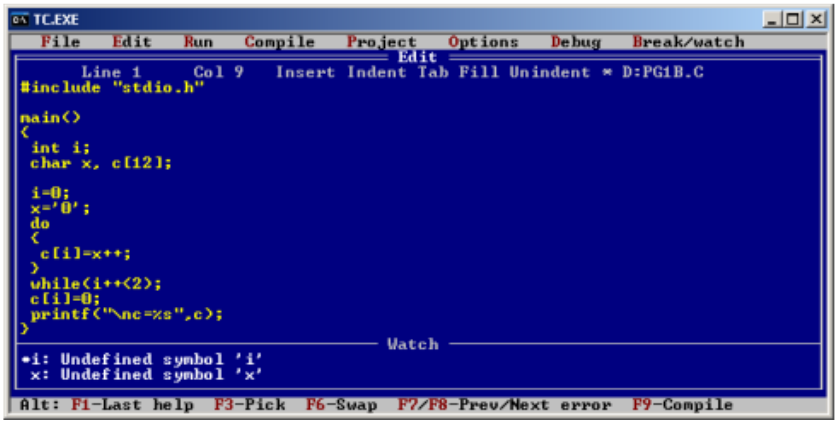
Что отобразится на экране по завершении программы

а) =70

б) x=70

в) 70

1.89.



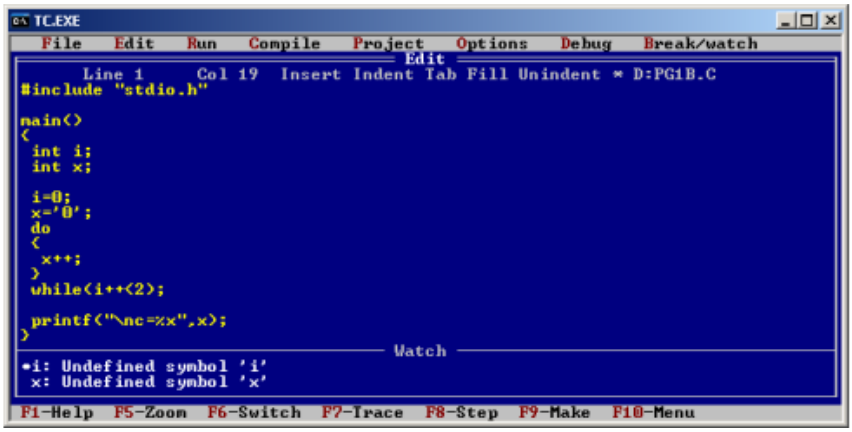
Что отобразится на экране по завершении программы

а) =012

б) c=012

в) 123

1.90.



Что отобразится на экране по завершении программы

а) C=33

б) C=51

в) 33

1.91.

11

```
char s1,s2[30];
int b;
b=s2[5];
```

В приведенном фрагменте программы (рисунок)
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка в первой строке

1.92.

12

```
char s1,s2[30];
int b;
b=int(s1);
```

В приведенном фрагменте программы (рисунок)
а) Ошибка в третьей строке
б) Нет синтаксических ошибок
в) Ошибка в первой строке

1.93.

13

```
char s1,s2[30];
int b;
s1+=s2[b];
```

В приведенном фрагменте программы (рисунок)
а) Ошибка в первой строке
б) Нет синтаксических ошибок
в) Ошибка в третьей строке

1.94.

14

```
char s1,s2[30];
int b;
s1%=s2[b++];
```

В приведенном фрагменте программы (рисунок)
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка в первой строке

1.95.

15

```
char s1,s2[30];
int b;
s1+=s1[20];
```

В приведенном фрагменте программы (рисунок)
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка в первой строке

1.96.

16

```
char s1,s2[30];
s1='d';
strcpy(s2,"d");
```

В приведенном фрагменте программы (рисунок)
а) Ошибка в первой строке
б) Нет синтаксических ошибок
в) Ошибка в третьей строке

1.97.

17

```
char s1,s2[30];
s1="d";
strcpy(s2,"d");
```

В приведенном фрагменте программы (рисунок)
а) Нет синтаксических ошибок
б) Ошибка во второй строке
в) Ошибка в первой строке

1.98.

18

```
char s1,s2[30];
s1='d';
strcpy(s2,'d');
```

В приведенном фрагменте программы (рисунок)
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка в первой строке

1.99.

19

```
char s1,s2[30];
s1='d';
s2="text";
```

В приведенном фрагменте программы (рисунок)
а) Ошибка в первой строке
б) Ошибка в третьей строке
в) Ошибка во второй строке

1.100.

20

```
char s1,s2[30];
s2[11]='d';
strcpy(s1,"d");
```

В приведенном фрагменте программы (рисунок)
а) Нет синтаксических ошибок
б) Ошибка в третьей строке
в) Ошибка в первой строке

2 Вопросы в открытой форме.

2.1. _____ _____– это совокупность следующих объектов: директив, указаний компилятору, объявлений и определений.

2.2. ____ – это язык программирования общего назначения, хорошо известный своей эффективностью, экономичностью, и переносимостью.

2.3. _____ _____– это последовательность букв, цифр и символов, заключенная в двойные кавычки. _____

2.4. Строчные литералы имеют тип _____.

2.5. _____ – это имена переменных, функций и меток, используемых в программе.

2.6. _____ – это единица текста программы, которая имеет определенный смысл для компилятора и которая не может быть разбита в дальнейшем.

2.7. _____ _____– это изменения состояния машины, которые возникают в результате вычисления выражений.

2.8. Си не поддерживает оператор "else if", но тот же самый эффект достигается посредством сложенных операторов _____.

2.9. _____ – это независимая совокупность объявлений и операторов, обычно предназначенная для выполнения определенной задачи.

2.10. Программы на Си состоят по крайней мере из одной функции ____, но могут содержать и больше функций.

2.11. _____ – это идентификатор, который может быть определен с квадратными скобками ([]), звездочкой (*) или круглыми скобками () для объявления массива, указателя или функции.

2.12. Спецификаторы типов float и double относятся к типу "_____".

2.13. _____ _____– это свойство, позволяющее выполнять программы на разных ЭВМ, работающих под управлением разных версий ОС UNIX, с минимальными изменениями.

2.14. 1TBS является аббревиатурой стиля _____.

2.15. В среде программирования Arduino исходный текст программы принято называть _____ , что можно перевести как «набросок, эскиз».

2.16. _____ – это миниатюрный компьютер, изготовляемый в виде печатной платы малых размеров.

2.17. Главным отличием Raspberry Pi от персонального компьютера является наличие на плате _____ и поддержка популярных интерфейсов для подключения периферийных устройств (UART (Serial); I2C/TWI; SPI).

2.18. На Raspberry Pi может быть установлена операционная система (ОС), поддерживающая ____ – процессоры.

2.19. ____ – это объектно-ориентированный, интерпретируемый, переносимый язык сверхвысокого уровня.

2.20. _____ содержит несколько элементов различных или одинаковых типов и может динамически изменять размер.

3. Вопросы на установление последовательности

3.1 Упорядочьте следующие действия в правильном порядке для вывода чисел от 1 до 10 с использованием цикла for:

а) Условие продолжения выполнения цикла;
б) Инкремент переменной счетчика;
в) Вывод текущего значения счетчика.
г) Инициализация переменной счетчика;

3.2 Расставьте этапы выполнения программы на микроконтроллере в правильном порядке:

а) Инициализация портов ввода-вывода.
б) Обработка данных и принятие решений.
в) Отправка данных на вывод.
г) Ожидание событий или внешних сигналов.
д) Чтение данных с датчика.

3.3 Установите правильный порядок действий при разработке встроенной системы на микроконтроллере:
а) Определение требований к системе.
б) Проектирование аппаратной части.
в) Тестирование и верификация системы.
г) Массовое производство и внедрение.
д) Написание и отладка программного кода.

3.4 Установите правильный порядок этапов разработки приложения для мобильного устройства:
а) Определение функциональных требований и интерфейса приложения.
б) Написание и отладка программного кода.
в) Тестирование на различных устройствах и ОС.
г) Подготовка к публикации в магазине приложений.
д) Проектирование пользовательского интерфейса.

3.5 Установите правильный порядок выполнения этапов для взаимодействия с датчиком света и светодиодом на платформе Arduino:
а) Чтение данных с датчика света и сохранение их в переменной.
б) Определение условия (например, если яркость меньше определенного значения).
в) Включение или выключение светодиода в зависимости от условия.
г) Запуск бесконечного цикла в функции void loop() для постоянного мониторинга и реагирования на изменения яркости.
д) Подключение датчика света к платформе Arduino и указание используемого порта.
е) Инициализация датчика света и светодиода в функции void setup().
ж) Включение и настройка датчика света для считывания данных о яркости.

3.6 Расставьте шаги в правильном порядке для начала работы с Raspberry Pi:
а) Вставка microSD-карты с операционной системой в слот на Raspberry Pi.
б) Подключение клавиатуры и мыши к USB-портам Raspberry Pi.
в) Подключение Raspberry Pi к источнику питания (например, микро-USB).
г) Подключение HDMI-кабеля к монитору и Raspberry Pi (если нужен вывод на монитор).
д) Подключение кабеля Ethernet (или настройка Wi-Fi, если необходимо).
е) Включение Raspberry Pi, либо путем включения питания, либо нажатием на кнопку питания (в зависимости от модели).

3.7 Установите правильный порядок для управления Raspberry Pi удаленно через SSH:
а) Получение IP-адреса Raspberry Pi в локальной сети.
б) Открытие терминала на компьютере и ввод команды SSH с указанием IP-адреса Raspberry Pi.
в) Ввод пароля или ключа SSH для аутентификации.
г) Успешное подключение к Raspberry Pi через SSH и возможность управления им удаленно.

д) Включение SSH на Raspberry Pi (если не активировано).

3.8 Расставьте шаги в правильном порядке для установки операционной системы на Raspberry Pi с использованием карты microSD:

а) Выбор образа операционной системы и целевого устройства (microSD-карты).

б) Начало процесса записи образа на microSD-карту.

в) Загрузка операционной системы Raspberry Pi на компьютер.

г) Запуск программы для записи образа операционной системы на microSD-карту (например, Etcher).

д) Извлечение microSD-карты из компьютера и вставка её в Raspberry Pi.

е) Подключение Raspberry Pi к источнику питания и включение его.

3.9 Расставьте шаги в правильном порядке для настройки доступа к веб-интерфейсу (веб-панели) Raspberry Pi:

а) Настройка веб-сервера для прослушивания на нужном порту и директории.

б) Публикация веб-страницы или приложения в директории веб-сервера.

в) Установка веб-сервера (например, Apache или Nginx) на Raspberry Pi.

г) Написание веб-страницы или приложения, которые будут отображаться через веб-интерфейс.

д) Открытие браузера на компьютере и ввод IP-адреса Raspberry Pi для доступа к веб-интерфейсу.

3.10 Расставьте этапы компиляции программы на языке C в правильном порядке:

а) Препроцессинг.

б) Компиляция.

в) Ассемблирование.

г) Линковка.

3.11 Установите последовательность выполнения операторов в языке C:

а) Условные операторы (if, else).

б) Циклы (for, while).

в) Операции присваивания.

г) Арифметические операции.

3.12 Установите последовательность разработки аппаратной части в инфокоммуникационных системах:

а) Проектирование схемы.

б) Печатная плата и монтаж.

в) Использование микроконтроллеров.

г) Сборка корпуса.

3.13 Расставьте этапы разработки программы для Arduino в правильном порядке:

а) Подключение к плате Arduino.

б) Загрузка программы на плату.

в) Написание кода в среде Arduino IDE.

г) Загрузка библиотек, если необходимо.

3.14 Установите последовательность выполнения цикла программы на Arduino:

а) Инициализация

б) Выполнение

в) Проверка условия

г) Обновление переменных

3.15 Расставьте этапы настройки Raspberry Pi в правильном порядке:

а) Подключение к Wi-Fi сети
б) Загрузка операционной системы на SD-карту
в) Подключение к монитору и клавиатуре
г) Установка необходимых программ и библиотек

3.16 Установите последовательность выполнения задачи на Raspberry Pi с использованием Python:

а) Импорт необходимых модулей
б) Запуск основного кода
в) Обработка данных или выполнение задачи
г) Завершение

3.17 В какой последовательности происходит процесс линковки программы на Си?

а) Обработка символов
б) Разрешение внешних ссылок
в) Подключение стандартных библиотек
г) Подключение пользовательских библиотек

3.18 В каком порядке вызываются функции стандартной библиотеки языка Си при выполнении программы?

а) main()
б) Математические функции
в) Функции ввода/вывода
г) Функции обработки строк
д) Функции управления памятью

3.19 В каком порядке происходит создание и уничтожение объектов в программе на Си с использованием конструкторов и деструкторов?

а) Создание объектов
б) Вызов конструкторов
в) Вызов деструкторов
г) Уничтожение объектов

3.20 В каком порядке располагаются элементы в структуре данных в языке Си?

а) Определение структуры
б) Инициализация структуры
в) Доступ к элементам структуры
г) Модификация элементов структуры
д) Уничтожение структуры

4 Вопросы на установление соответствия.

4.1 Установить соответствие основных операторов языка С с их описанием:

1) if-else	a) Циклический оператор, который выполняет определенный блок кода до тех пор, пока условие истинно.
2) for	b) Оператор выбора, который позволяет выбирать выполнение кода в зависимости от значения выражения.
3) while	c) Оператор выбора, который позволяет выполнить различный набор инструкций в зависимости от условия.

4) switch	d) Циклический оператор, используемый для выполнения определенного блока кода заданное количество раз.

4.2 Установить соответствие компонентов аппаратно-программных платформ с их описанием:

1) Центральный процессор (CPU)	a) Основной вычислительный элемент компьютера, который выполняет инструкции программы.
2) Ввод-вывод (I/O) порты	b) Временное хранилище данных и программ, используемое компьютером при выполнении задач.
3) Оперативная память (RAM)	c) Интерфейсы, которые позволяют компьютеру обмениваться данными с внешними устройствами.
4) Флэш-память (Flash Memory)	d) Постоянное хранилище данных и программ, которое сохраняет информацию после выключения питания.

4.3 Установить соответствие элементов архитектуры Arduino с их функциями:

1) Микроконтроллер ATmega328P	a) Постоянное хранилище данных, которое можно использовать для сохранения информации между запусками программы.
2) Цифровые пины ввода/вывода	b) Пины, предназначенные для аналогового ввода, позволяющие считывать аналоговые значения, например, с датчиков.
3) Аналоговые пины ввода	c) Пины, которые могут работать как цифровые входы или выходы для подключения и управления различными устройствами.
4) EEPROM (Электрически стираемое и программируемое постоянное запоминающее устройство)	d) Постоянное хранилище данных, которое можно использовать для сохранения информации между запусками программы.

4.4 Установить соответствие элементов архитектуры Raspberry Pi с их функциями:

1) Оперативная память (RAM)	a) Временное хранилище данных и программ, используемое Raspberry Pi при выполнении задач.
2) Центральный процессор (CPU)	b) Основной вычислительный элемент Raspberry Pi, который выполняет операционную систему и программы.
3) HDMI порт	c) Пины, которые могут использоваться для подключения и управления внешними устройствами, такими как сенсоры и моторы.
4) GPIO (General Purpose Input/Output) пины	d) Порт, который позволяет подключить Raspberry Pi к монитору или телевизору для вывода видео и аудио сигнала.

4.5 Установить соответствие типов данных в языке C с их описанием:

1) int	a) Целочисленный тип данных, который используется для хранения целых чисел.
2) char	b) Тип данных, предназначенный для хранения символов, таких как буквы и знаки пунктуации.
3) float	c) Тип данных с плавающей точкой, предназначенный для хранения чисел с плавающей

	запятой одинарной точности.
4) double	d) Тип данных с плавающей точкой двойной точности, предназначенный для хранения чисел с высокой точностью.

4.6 Установить соответствие:

1) Массив	a) Упорядоченный набор данных одного типа, которые могут быть доступны по индексу.
2) Строка (массив символов)	b) Массив символов, который представляет собой последовательность символов, завершающуюся нулевым символом.
3) Индекс	c) Число, которое используется для указания позиции элемента в массиве или строке.
4) Элемент	d) Конкретное значение в массиве или строке, находящееся на определенной позиции.

4.7 Установить соответствие:

1) Указатель	a) Переменная, которая содержит адрес в памяти другой переменной.
2) Адрес переменной	b) Уникальное значение, которое указывает на местонахождение переменной в памяти.
3) Разыменование указателя	c) Операция, которая позволяет получить значение, на которое указывает указатель.
4) Арифметика указателей	d) Манипуляции с указателями, такие как сложение или вычитание числа, которое влияет на текущий адрес указателя.

4.8 Установить соответствие:

1) if	a) Оператор выбора, который выполняет код, если условие истинно, иначе переходит к другой ветке.
2) else	b) Оператор, который выполняется, если условие в старшем операторе ложно.
3) switch	c) Конструкция выбора, которая позволяет выбирать выполнение кода в зависимости от значения выражения.
4) ? : (тернарный оператор)	d) Тернарный оператор, который выполняет одно из двух выражений в зависимости от значения условия.

4.9 Установить соответствие:

1) Указатель на функцию	a) Тип данных, который может хранить адрес функции и позволяет вызывать функцию, на которую он указывает.
2) Обратный вызов (callback)	b) Механизм, который позволяет передавать функцию как аргумент другой функции и вызывать ее позднее.
3) Функция высшего порядка	c) Функция, которая может принимать другие функции в качестве аргументов или возвращать их как результат.
4) Указатель на член структуры	d) Указатель, который указывает на конкретное поле структуры и позволяет получить доступ к нему.

4.10 Установить соответствие:

1) fopen()	a) Функция, которая открывает файл для чтения или записи и возвращает указатель на файл.
2) fclose()	b) Функция, которая закрывает открытый файл и освобождает ресурсы.
3) fread()	c) Функция, которая считывает данные из файла и сохраняет их в буфере.
4) fwrite()	d) Функция, которая записывает данные из буфера в файл.

4.11 Установите соответствие:

1) #include	a) Директива, которая включает содержимое указанного файла в исходный код программы.
2) #define	b) Директива, которая используется для создания макросов и замены их в коде программы.
3) #ifdef	c) Директива, которая проверяет, определен ли макрос препроцессора, и выполняет код в зависимости от результата.
4) #ifndef	d) Директива, которая проверяет, не определен ли макрос препроцессора, и выполняет код в зависимости от результата.

4.12 Установить соответствие:

1) errno	a) Глобальная переменная, которая содержит код ошибки после выполнения некоторых системных функций.
2) perror()	b) Функция, которая выводит сообщение об ошибке, соответствующее текущему значению errno.
3) try-catch	c) Конструкция, используемая для обработки исключительных ситуаций и ошибок в программе.
4) setjmp() и longjmp()	d) Функции, которые позволяют реализовать механизм низкоуровневой обработки исключений в C.

4.13 Установить соответствие терминов и понятий, связанных с многопоточным программированием в языке C, с их описанием:

1) Поток (thread)	a) Выполняющаяся часть программы, которая может работать параллельно с другими потоками.
2) Синхронизация	b) Координация выполнения потоков, чтобы избежать гонок данных и других проблем многопоточности.
3) Критическая секция	c) Участок кода, который должен выполняться только одним потоком в данный момент времени.
4) Мьютекс (mutex)	d) Объект, используемый для блокировки доступа к ресурсам, позволяя только одному потоку использовать их в определенный момент времени.

4.14 Установить соответствие функций и библиотек языка C с их описанием:

1) printf()	a) Функция, используемая для вывода данных на консоль.
2) scanf()	b) Функция, используемая для чтения данных с консоли.
3) stdlib.h	c) Библиотека, которая содержит функции для работы с памятью и другими стандартными функциями.

4) string.h	d) Библиотека, которая содержит функции для работы со строками, включая копирование, сравнение и другие операции.

4.15 Установить соответствие:

1) Перечисление (enum)	a) Специальный тип данных, который представляет собой набор именованных целочисленных констант.
2) Перечислимый тип (enum type)	b) Абстрактный тип данных, который может содержать набор констант из перечисления.
3) Элемент перечисления	c) Каждый элемент в перечислении, которому присвоено уникальное целочисленное значение.
4) Инициализация перечисления	d) Процесс задания значений для элементов перечисления при его создании.

4.16 Установить соответствие:

1) malloc()	a) Функция, которая заполняет блок памяти указанным значением.
2) free()	b) Функция, которая копирует данные из одного блока памяти в другой.
3) memset()	c) Функция, которая используется для выделения блока динамической памяти заданного размера в куче.
4) memcpy()	d) Функция, которая освобождает ранее выделенную динамическую память.

4.17 Установить соответствие:

1) Ethernet кабель (CAT6)	a) Используется для передачи данных в высокоскоростных сетях, таких как Gigabit Ethernet.
2) Витая пара (Twisted Pair)	b) Используется для соединения компьютеров в локальной сети.
3) Оптоволоконный кабель (Fiber Optic)	c) Обеспечивает высокую пропускную способность и долгие расстояния передачи данных.
4) Коаксиальный кабель (Coaxial)	d) Ранее часто использовался для подключения телевизионных сетей.

4.18 Установить соответствие:

1) Raspbian (теперь известная как Raspberry Pi OS)	a) Оптимизирована для мультимедийного воспроизведения.
2) Ubuntu Mate	b) Основана на Debian Linux.
3) OSMC (Open Source Media Center)	c) Предназначена для создания игровых консолей.
4) RetroPie	d) Имеет графический интерфейс, подходящий для начинающих.

4.19 Установить соответствие:

1) PyCharm	a) Интерактивная среда для выполнения кода Python по строкам, обычно используется для анализа данных.

2) Visual Studio Code	b) Простая среда Python, предустановленная с Python, которую можно использовать для быстрых экспериментов.
3) Jupyter Notebook	c) Интегрированная среда разработки (IDE) для Python с множеством функций и поддержкой сторонних плагинов.
4) IDLE	d) Легковесная среда с расширяемым функционалом и хорошей поддержкой Python.

4.20 Установить соответствие:

1) print()	a) Используется для вывода текста или данных на экран.
2) input()	b) Используется для получения пользовательского ввода с клавиатуры.
3) open()	c) Используется для открытия файла в режиме чтения или записи.
4) close()	d) Используется для закрытия открытого файла.

Шкала оценивания результатов тестирования: в соответствии с действующей в университете балльно-рейтинговой системой оценивание результатов промежуточной аттестации обучающихся осуществляется в рамках 100-балльной шкалы, при этом максимальный балл по промежуточной аттестации обучающихся по очной форме обучения составляет 36 баллов (установлено положением П 02.016).

Максимальный балл за тестирование представляет собой разность двух чисел: максимального балла по промежуточной аттестации для данной формы обучения (36) и максимального балла за решение компетентностно-ориентированной задачи (12).

Балл, полученный обучающимся за тестирование, суммируется с баллом, выставленным ему за решение компетентностно-ориентированной задачи.

Общий балл по промежуточной аттестации суммируется с баллами, полученными обучающимся по результатам текущего контроля успеваемости в течение семестра; сумма баллов переводится в оценку по 5-балльной шкале следующим образом:

Соответствие 100-балльной и 5-балльной шкал

Сумма баллов по 100-балльной шкале	*Оценка по 5-балльной шкале*
100-85	отлично
84-70	хорошо
69-50	удовлетворительно
49 и менее	неудовлетворительно

Критерии оценивания результатов тестирования:

Каждый вопрос (задание) в тестовой форме оценивается по дихотомической шкале: выполнено – **2 балла**, не выполнено – **0 баллов**.

2.2 КОМПЕТЕНТНОСТНО-ОРИЕНТИРОВАННЫЕ ЗАДАЧИ

Компетентностно-ориентированная задача № 1

Напишите программу, которая вычисляет сумму всех целых чисел от 1 до 100 и выводит результат на экран. Для этой задачи вы можете использовать цикл, например, **for**, чтобы последовательно пройти по числам от 1 до 100 и накапливать сумму.

Компетентностно-ориентированная задача № 2

Разработайте программу, которая запрашивает у пользователя целое число и затем

вычисляет факториал этого числа. Факториал числа n - это произведение всех целых чисел от 1 до n.

Компетентностно-ориентированная задача № 3

Создайте программу, которая проверяет, является ли введенное пользователем число простым. Простое число - это число, которое делится только на 1 и само себя.

Компетентностно-ориентированная задача № 4

Напишите программу, которая позволяет пользователю ввести температуру в градусах Цельсия, а затем конвертирует её в градусы Фаренгейта и наоборот, используя формулы для преобразования температур.

Компетентностно-ориентированная задача № 5

Реализуйте программу, которая запрашивает у пользователя два целых числа и находит их наибольший общий делитель (НОД) с использованием алгоритма Евклида.

Компетентностно-ориентированная задача № 6

Создайте программу, которая сортирует массив целых чисел в порядке возрастания, используя алгоритм сортировки пузырьком. Алгоритм пузырьковой сортировки сравнивает пары соседних элементов и меняет их местами, если они находятся в неправильном порядке, продолжая проход по массиву до тех пор, пока весь массив не будет отсортирован.

Компетентностно-ориентированная задача № 7

Разработайте программу, которая проверяет, является ли введенная строка палиндромом. Палиндром - это строка, которая читается одинаково как с начала, так и с конца, игнорируя пробелы и знаки препинания.

Компетентностно-ориентированная задача № 8

Напишите программу, которая позволяет пользователю ввести коэффициенты квадратного уравнения (a, b и c), а затем находит его корни с использованием формулы квадратного корня.

Компетентностно-ориентированная задача № 9

Реализуйте программу, которая находит числа Фибоначчи до заданного числа n с использованием цикла. Числа Фибоначчи - это последовательность чисел, в которой каждое число равно сумме двух предыдущих чисел (например, 0, 1, 1, 2, 3, 5, 8 и так далее).

Компетентностно-ориентированная задача № 10

Создайте программу для чтения данных из текстового файла, их обработки (например, вычисление среднего значения) и записи результатов в другой файл. Эта задача поможет вам овладеть работой с файлами в языке C.

Компетентностно-ориентированная задача № 11

Разработайте программу-калькулятор, которая предлагает пользователю ввести два числа и выбрать операцию (сложение, вычитание, умножение или деление). Затем программа выполняет выбранную операцию и выводит результат.

Компетентностно-ориентированная задача № 12

Напишите программу, которая находит и выводит все простые числа в заданном диапазоне, введенном пользователем. Простые числа - это натуральные числа, большие 1, которые делятся только на 1 и самих себя.

Компетентностно-ориентированная задача № 13

Разработайте программу для чтения данных из CSV-файла, где значения разделены запятыми. После чтения файла программа вычисляет среднее значение чисел в каждом столбце и выводит результаты.

Компетентностно-ориентированная задача № 14

Напишите программу, которая генерирует случайные числа, вычисляет среднее, минимальное и максимальное значение из сгенерированных чисел, а также подсчитывает, сколько чисел было сгенерировано в заданном диапазоне.

Компетентностно-ориентированная задача № 15

Разработайте программу, которая анализирует текстовый файл и выводит статистику, такую как количество слов, предложений, символов.

Компетентностно-ориентированная задача № 16

Напишите программу, которая моделирует работу банкомата. Пользователь может вводить сумму для снятия, и программа должна проверить, есть ли достаточное количество средств на счете, и, если да, выдать необходимое количество купюр.

Компетентностно-ориентированная задача № 17

Разработайте программу для генерации случайных паролей заданной длины. Пароль может включать буквы, цифры и специальные символы. Пользователь указывает желаемую длину пароля, а программа генерирует случайный пароль и выводит его.

Компетентностно-ориентированная задача № 18

Создайте схему подключения текстового LCD-дисплея к Arduino.

Компетентностно-ориентированная задача № 19

Разработайте схему подключения сенсора DHT к Arduino.

Компетентностно-ориентированная задача № 20

написать и отладить программу, запрашивающую в бесконечном диалоге у пользователя операнд, возводящую полученное значение в квадрат и выводящую результат на экран. Требуется обеспечить вывод их диалога.

Компетентностно-ориентированная задача № 21

Написать и отладить программу, перемещающую в бесконечном диалоге с пользователем символьную строку на экране.

Направление перемещения определяется нажатием клавиш:

- 1 – влево;
- 2 - вправо;
- 3 -вверх;
- 4 - вниз.

Требуется обеспечить выход их диалога.

Компетентностно-ориентированная задача № 22

Написать и отладить программу, выводящую график некоторой математической функции sin на экран в графическом режиме.

Компетентностно-ориентированная задача № 23

Написать и отладить программу, выводящую график некоторой математической

функции (логарифм) на экран в графическом режиме.

Компетентностно-ориентированная задача № 24

Написать и отладить программу, выводящую график некоторой математической функции cos на экран в графическом режиме.

Компетентностно-ориентированная задача № 25

Разработайте простую игру, в которой компьютер "загадывает" случайное число в определенном диапазоне, а игрок должен угадать это число. Программа должна предоставлять подсказки о том, было ли предложенное число слишком маленьким или слишком большим, и затем позволить игроку угадать число с ограниченным количеством попыток.

Шкала оценивания решения компетентностно-ориентированной задачи*:* в соответствии с действующей в университете балльно-рейтинговой системой оценивание результатов промежуточной аттестации обучающихся осуществляется в рамках 100-балльной шкалы, при этом максимальный балл по промежуточной аттестации обучающихся по очной форме обучения составляет 36 баллов (установлено положением П 02.016).

Максимальное количество баллов за решение компетентностно-ориентированной задачи – 12 баллов. Балл, полученный обучающимся за решение компетентностно-ориентированной задачи, суммируется с баллом, выставленным ему по результатам тестирования.

Общий балл промежуточной аттестации суммируется с баллами, полученными обучающимся по результатам текущего контроля успеваемости в течение семестра; сумма баллов переводится в оценку по 5-баллной шкале следующим образом*:*

Сумма баллов по 100-балльной шкале	*Оценка по 5-балльной шкале*
100-85	отлично
84-70	хорошо
69-50	удовлетворительно
49 и менее	неудовлетворительно

Критерии оценивания решения компетентностно-ориентированной задачи

12-10 баллов выставляется обучающемуся, если решение задачи демонстрирует глубокое понимание обучающимся предложенной проблемы и разностороннее ее рассмотрение; работа представляет собой логичное, ясное и при этом краткое, точное описание хода решения задачи (последовательности (или выполнения) необходимых трудовых действий) и формулировку доказанного, правильного вывода (ответа); при этом обучающимся предложено несколько вариантов решения или оригинальное, нестандартное решение (или наиболее эффективное, или наиболее рациональное, или оптимальное, или единственно правильное решение); задача решена в установленное преподавателем время или с опережением времени.

9-7 балла выставляется обучающемуся, если решение задачи демонстрирует понимание обучающимся предложенной проблемы; задача решена типовым способом в установленное преподавателем время; имеют место общие фразы и (или) несущественные недочеты в описании хода решения и (или) вывода (ответа).

6-4 балла выставляется обучающемуся, если решение задачи демонстрирует поверхностное понимание обучающимся предложенной проблемы; осуществлена попытка шаблонного решения задачи, но при ее решении допущены ошибки и (или) превышено установленное преподавателем время.

0-3 баллов выставляется обучающемуся, если решение задачи демонстрирует

непонимание обучающимся предложенной проблемы, и (или) значительное место занимают общие фразы и голословные рассуждения, и (или) задача не решена.