Согласовано
Начальник отдела эксплуатации и
внедрения информационных систем
ОГАУЗ СОМИАЦ
_____ Я.А.Комиссаров
«31» 08 2020 г.

УТВЕРЖДАЮ
Заместитель директора по
учебной работе
_____ И. В. Иванешко
«31» 08 2020 г.

Контрольно-оценочные материалы для промежуточной аттестации по общепрофессиональной дисциплине
ОП.05. Основы программирования
Для специальности 09.02.03 Программирование в компьютерных системах

Экзамен является промежуточной формой контроля, подводит итог освоения дисциплины ОП.05. Основы программирования.

В результате освоения дисциплины студент должен освоить следующие профессиональные компетенции:

ПК 1.1. Выполнять разработку спецификаций отдельных компонент

ПК 1.2. Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля

ПК 1.3. Выполнять отладку программных модулей с использованием специализированных программных средств

ПК 1.4. Выполнять тестирование программных модулей

ПК 1.5. Осуществлять оптимизацию программного кода модуля

ПК 3.1. Анализировать проектную и техническую документацию на уровне взаимодействия компонент программного обеспечения

А также общие компетенции:

ОК 1. Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.

ОК 2. Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК 3. Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.

ОК 4. Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК 5. Использовать информационно – коммуникационные технологии в профессиональной деятельности.

ОК 6. Работать в коллективе и команде, эффективно общаться с коллегами, руководством, потребителями.

ОК 7. Брать на себя ответственность за работу членов команды (подчиненных), за результат выполнения заданий.

ОК 8. Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК 9. Ориентироваться в условиях частой смены технологий в профессиональной деятельности.

В ходе экзамена проверяется сформированность:

Знаний:

31 этапы решения задачи на компьютере;

32 типы данных;

33 базовые конструкции изучаемых языков программирования;

34 принципы структурного и модульного программирования;

35 принципы объектно-ориентированного программирования.

36 Обзор и область применения языков программирования;

З7 Библиотеки шаблонов и классов;

З8 Исключительные ситуации и ошибки;

З9 Тестирование и отладка программы;

З10 Стили и культура программирования.

Умений:

У1 работать в среде программирования;

У2 реализовывать построенные алгоритмы в виде программ на конкретном языке программирования.

У3 Выполнять тестирование и отладку программы;

У4 Обрабатывать ошибки и исключительные ситуации;

У5 Применять объекты готовых шаблонов классов.

Экзамен проводится в форме тестирования. Тест содержит 360 вопросов (суммарно тестовых позиций с выбором ответа и теоретических вопросов с ответом), выбираемых случайным образом программой из блоков заданий по каждой компетенции - ПК.1.1 - 5 вопросов (3 с выбором ответа, 2 с ответом), ПК.1.2 – 5 вопросов (3 с выбором ответа, 2 с ответом), ПК.1.3 – 5 вопросов (3 с выбором ответа, 2 с ответом), ПК.1.4 – 5 вопросов (3 с выбором ответа, 2 с ответом), ПК.1.5 - 5 вопросов (3 с выбором ответа, 2 с ответом), ПК.3.1 – 5 вопросов (3 с выбором ответа, 2 с ответом). Итого будет выбрано 18 вопросов с выбором ответа и 12 вопросов с ответом.

Время тестирования – 72 минуты (по 2 минуты на вопрос с выбором ответа, 3 минуты на вопрос с ответом).

Критерии оценивания

«5» - соответствует работа, содержащая 100-90% правильных ответов;

«4» - соответствует работа, содержащая 75-89% правильных ответов;

«3» - соответствует работа, содержащая 60-74% правильных ответов;

«2» - соответствует работа, содержащая менее 60% правильных ответов.

Шкала оценивания образовательных результатов:

Оценка	Критерии
«отлично»	Студент набрал 5 баллов
«хорошо»	Студент набрал 4 балл
«удовлетворительно»	Студент набрал 3 балла
«неудовлетворительно»	Студент набрал 0-2 балла

Первый блок заданий – вопросы с выбором ответа:

Проверяемая компетенция - ПК 1.1.
Проверяемые общие компетенции - ОК1-ОК9

1. Что представляет собой программа в технологии структурного программирования?
   a) последовательность действий
   b) несколько взаимодействующих объектов
   c) система определений функций, описание того, что нужно вычислить

2. Что представляет собой программа в технологии объектно-ориентированного программирования?
   a) последовательность действий
   b) несколько взаимодействующих объектов
   c) система определений функций, описание того, что нужно вычислить

3. Что представляет собой программа в технологии функционального программирования?
   a) последовательность действий
   b) несколько взаимодействующих объектов
   c) система определений функций, описание того, что нужно вычислить

4. Какие принципы относятся к принципам структурного программирования?
   a) отказ от оператора безусловного перехода
   b) объединение данных и алгоритмов их обработки в один объект
   c) любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл
   d) декларирование целей программы вместо алгоритма

5. Каковы достоинства структурного программирования?
   a) Удобочитаемость программ
   b) Упрощение отладки и тестирование программ
   c) Широчайшие возможности для автоматического распараллеливания вычислений
   d) Уменьшается количество кода

6. Цикл какого типа должен выполниться хотя бы один раз?
   a) Цикл с предусловием;
   b) Цикл с постусловием;
   c) Цикл со счетчиком;
   d) Все циклические конструкции;

7. Каковы особенности цикла со счетчиком?
   a) Выполняется заданное количество раз;
   b) Выполняется, пока истинное условие;
   c) Не является универсальным;
   d) Является универсальным;

8. Какие существуют виды цикла?
   a) С подусловием;
   b) С предусловием;
   c) С заданным количеством повторений;
   d) С постепенным условием;

9. Какова особенность цикла с постусловием?
   a) Должен выполниться хотя бы один раз;
   b) Не выполнится ни разу;

c) Выполнится один раз;
d) Выполнится заданное количество раз;

10. Что такое сортировка?
a) Упорядочение данных
b) Выбор только положительных элементов
c) Нумерация элементов массива с 1

11. Что выполнит данный псевдокод программы?

```
For I = 0 To 3 do
 For J = 0 To 4 - i do
  If M[J]> M[J + 1] Then
   Begin
   Tmp: = M[J];
    M[J]: = M[J + 1];
    M[J + 1]: = Tmp;
  End;
```

a) Сортирует массив по убыванию методом прямого выбора
b) Сортирует массив по возрастанию методом пузырька
c) Меняет порядок элементов массива на обратный
d) Сортирует массив по убыванию методом пузырька

12. На каком этапе жизненного цикла программы пишется техническое задание?
a) Маркетинг и спецификация программного продукта
b) Проектирование структуры программного продукта
c) Документирование программного продукта
d) Программирование

13. На каком этапе решения задачи определяется форма выдачи результатов?
a) Постановка задачи
b) Анализ и исследование задачи
c) Разработка алгоритма
d) Программирование
e) Тестирование и отладка

14. В чем сущность концепции модульного программирования?
a) в разбиении программы на отдельные функционально независимые части;
b) в разбиении программы на отдельные равные части;
c) в разбиение программы на процедуры и функции;

15. Каковы рекомендуемые размеры модулей программы?
a) небольшие;
b) большие;
c) равные;
d) фиксированной длины.

16. В чем заключается независимость программного модуля?
a) в написании, отладке и тестировании независимо от остальных модулей;
b) в разработке и написании независимо от других модулей;
c) в независимости от работы основной программы.

17. Является ли оператор множественного ветвления базовой алгоритмической структурой?

a) Да
b) Нет

18. Какой вид цикла присутствует в данном фрагменте программы?

```
int i=0;
int y=0;
do
 {   i++;
    y=y+1/i;}
while (1/i<e);
```

a) Цикл с предусловием;
b) Цикл с постусловием;
c) Цикл со счетчиком;
d) Тут нет циклических алгоритмов;

19. Какой вид цикла присутствует в данном фрагменте программы?

```
y=1; k=0;
while (y<=M)
{
  y=y*3;
  k++;
}
```

a) Цикл с предусловием;
b) Цикл с постусловием;
c) Цикл со счетчиком;
d) Нет циклических конструкций;

20. Какие параметры подпрограммы называются формальными?
a) Указанные при вызове функции реальные значения;
b) Локальные переменные;
c) Указанные переменные в заголовке функции при ее описании;

21. Какие параметры подпрограммы называются фактичекими?
a) Локальные переменные;
b) Указанные переменные в заголовке функции при ее описании;
c) Указанные при вызове функции реальные значения;

22. Сколько раз можно вызвать подпрограмму в программе?
a) Только 1 раз;
b) Только 10 раз;
c) Столько, сколько необходимо раз;

23. Как можно вызвать функцию в программе?
a) Указанием имени;
b) Указанием и имени и фактических параметров в скобках, если они нужны;
c) Указанием и имени и формальных параметров в скобках;

24. Какие элементы может содержать тип Массив?
a) Только однотипные элементы;
b) Разнотипные элементы;
c) Только строковую информацию

25. Каковы основные принципы объектно-ориентированного программирования?
a) Наследование

b) Полиморфизм
c) Полиформизм
d) Инклюдирование
e) Инкапсуляция

26. Какое понятие более общее?
    a) Класс
    b) Объект
    c) Конструктор
    d) Атрибуты класса

27. Как называется свойство родственных классов решать сходные проблемы различными способами (алгоритмами)?
    a) Наследование
    b) Полиморфизм
    c) Полиформизм
    d) Инклюдирование
    e) Инкапсуляция

28. Как называется свойство объектов порождать своих потомков?
    a) Наследование
    b) Полиморфизм
    c) Полиформизм
    d) Инклюдирование
    e) Инкапсуляция

29. Как называется относительно самостоятельная часть программы, имеющая свое назначение и реализующая некий алгоритм?
    a) Подпрограмма
    b) Массив
    c) Структура
    d) Файл

30. Как называется объединение в единое целое данных и алгоритмов обработки этих данных?
    a) Наследование
    b) Полиморфизм
    c) Полиформизм
    d) Инклюдирование
    e) Инкапсуляция

Проверяемая компетенция - ПК 1.2.
Проверяемые общие компетенции - ОК1-ОК9

1. Какой вид цикла присутствует в данном фрагменте программы?

```
int i=0;
int y=0;
do
 {  i++;
   y=y+1/i;}
while (1/i<e);
```

    a) Цикл с предусловием;
    b) Цикл с постусловием;

c) Цикл со счетчиком;
d) Тут нет циклических алгоритмов;

2. Какой вид цикла присутствует в данном фрагменте программы?

```
y=1; k=0;
while (y<=M)
{
  y=y*3;
  k++;
}
```

a) Цикл с предусловием;
b) Цикл с постусловием;
c) Цикл со счетчиком;
d) Нет циклических конструкций;

3. Какие типы в языке C++ целыми типами данных?
   a) int
   b) float
   c) short
   d) integer
   e) unsigned int

4. Какие имена могут использоваться в качестве идентификаторов при написании программ в языке C++?
   a) x1
   b) 1x
   c) for
   d) AAA

5. Как в языках C-семейства может выглядеть инкремент переменной x?
   a) x:=x+1
   b) x=x+1
   c) x++
   d) x=y+1
   e) x--
   f) ++x

6. Какие операторы относятся к операторам ветвления в языке C++?
   a) Оператор switch
   b) Оператор while
   c) Оператор break
   d) Оператор if

7. Что выполняет данный фрагмент программы?

```
int i,S;
S=0;
i=1;
do
{ S=S+i*2;
i++; }
while (2*i<100);
}
```

a) Подсчет суммы квадратов чисел от 1 до 100;

b) Подсчет суммы ряда натуральных чисел от 1 до 100;
c) Подсчет суммы четных чисел от 1 до 100;
d) Программа содержит ошибку;

8. Что позволяет выполнить команда continue позволяет в языках программирования С-семейства?
a) Досрочно выйти из цикла
b) Досрочно выйти из программы
c) Выйти из текущей итерации цикла и продолжить цикл дальше;
d) Начать цикл заново

9. Какое действие выполнит метод класса string insert(i,st) в языке С++?
a) вставит в строку s начиная с позиции i строку st
b) вставит в строку st начиная с позиции i строку s
c) вставит в строку i начиная с позиции st строку s
d) вставит в строку i начиная с позиции s строку st

10. Задан класс:

```
class Complex
{ int real;
  int im;
  void Add(Complex x);
  void Mult(Complex x);}
```

Что является атрибутами данного класса?
a) real
b) Add
c) im
d) Mult

11. Какой доступ к элементам класса дает спецификация доступа protected?
a) Доступ к элементам только в этом классе
b) Доступ к элементам из всех классов (общий)
c) Доступ к элементам из производных классов
d) Доступ к элементам из базового и производных классов

12. Дан фрагмент программы на языке С++:

```
int x;
int y;
cin>>x;
cin>>y;
y=x/y;
cout<<y;
```

Что будет содержать переменная y?
a) содержит результат деления x на y
b) содержит результат целочисленного деления x на y
c) фрагмент содержит ошибку из-за несоответствия типов

13. Дан фрагмент программы на языке C++:

```
char x;
int y;
cin>>x;
y=x;
cout<<y;
```

Что будет содержать переменная y?

a) содержать код символа x
b) фрагмент содержит ошибку из-за несоответствия типов
c) содержать символ x

14. Дан фрагмент программы на языке C++:

```
void main( )
{ int top;
top  = 5%2*10;
top  =top++;
cout<<"top =", top;
}
```

Что будет выведено в результате выполнения?

a) 11
b) 5
c) 6
d) 10
e) 21

15. Какое утверждение является верным?

a) if…else более универсальный оператор, чем switch
b) if…else менее универсальный оператор, чем switch
c) if…else выполняет то же самое, что и switch

16. Какие типы может иметь ключ выбора в операторе switch?

a) int
b) char
c) float
d) double

17. Дана функция:

```
int number(int a,int b)
 {int m;
  if (a>b) m=a;
else m=b;
  return m;}
```

Что вычислит эта функция?

a) максимальное из 2-х чисел;
b) Вычисляет минимальное из 2-х чисел;

18. Дана программа с использованием функции:

```
#include <iostream>;

int sum( int x )
{      if ( x==0 )
       return 0;
       else
  return sum(x-1)+x; }

void main()
{     int n;
    cin >> n;
    cout << "sum= " << sum(n) << endl;
    }
```

Что вычислит эта функция?

a) сумму чисел 1+n
b) сумму неотрицательных чисел 1+n
c) Сумму чисел от 1 до N
d) сумму ряда с заданной точностью

19. С какого значения в С-семействе языков программирования начинается нумерация элементов в массиве?

a) с 1
b) по желанию программиста
c) с 0

20. Дан следующий фрагмент программы на языке С++:

```
int m[5]={1,2,5,67,3};
int i;
for (i=1;i<10;i++)
if (mas[i]%2=0)
 cout<<mas[i]<<endl;
```

Что он выполнит?

a) Поиск всех нечетных элементов массива
b) Вывод всех элементов массива
c) Вывод всех четных элементов массива
d) Вывод половины элементов массива

21. Что выполнит данная программа?

```
#include <iostream>;
void main()
{int p,i=0;
int a[10]={10,11,12,13,14,15,16,17,18,19};
while(i<10/2)
{p=a[i];
a[i]=a[9-i];
a[9-i]=p;
i++;}
i=0;
while(i<10)
cout<<"  ”<<a[i++];}
```

a) Ищет сумму элементов массива
b) Меняет местами первый и последний элементы массива
c) Сортирует массив по возрастанию
d) Меняет порядок элементов массива на обратный

22. Каковы будут правильные обращения к элементу массива int mas[5]={1,2,3,4,5} на языке C++?

a) cout<<mas[5];
b) cout<<mas[4];
c) cout<<mas;
d) cout<<mas[0];

23. Что выполнит данный фрагмент программы?

```
for (i = 0; i < ARR_SIZE; i++)
for (j = 0; j < ARR_SIZE; j++)
    if (X[i][j] > 0) && (X[i][j] < Xmin)
        { Xmin = X[i][j];
          min = i; }
```

a) Находит наименьший положительный элемент массива и его индексы
b) Находит наименьший положительный элемент массива и номер его строки
c) Находит наименьший положительный элемент массива и номер его столбца
d) Находит наименьший элемент массива и номер его строки

24. Что выполнит данная программа?

```
#include <iostream>
#include <string.h>
using namespace std;

int main() {
char s[200];
char s1[200];
cin.getline(s,sizeof(s));
unsigned int i=0;
while(s[i]!='\0')
{if (s[i]!='  ')
  s1[i]=s[i];
i++;}
cout <<s1<< endl;
return 0;
}
```

a) подсчитывает в строке количество слов
b) подсчитывает в строке количество пробелов
c) заменяет все пробелы на 1
d) удаляет из строки все пробелы

25. Каково служебное слово для объявления структуры в языках С-семейства?

a) Record
b) struct
c) string
d) union

26. Что выполнит данная программа?

```
#include <iostream>
# include <string.h>
struct student
{    char name[30];
     int kurs;
     int age;};
void main()
{  struct student stud[10];
    int i, n;
    cout<<"Количество студентов:"<<endl;
    cin>>n;
    for(i=0;i<n;i++)
    {
cout<<"Введите имя:"<<endl;
cin>>stud[i].name;
cout<<"Введите возраст:"<< endl;
cin>>stud[i].age;
cout<<"Введите номер курса:"<<endl;
cin>>stud[i].kurs;
    }

for(i=0;i<n;i++)
  if (stud[i].kurs==4)
   cout<<stud[i].name<<"  "<<stud[i].age;
   }
```

a) Выполняет поиск четвертого номера студента
b) Выполняет поиск ударников
c) Выполняет поиск студентов, которым 4 года
d) Выполняет поиск четверокурсников

27. Каким словом обозначены элементы класса, которые должны быть доступны в классе и вне класса?

a) protected
b) public
c) private

28. Что выполнит данная программа?

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

void MulInt (int number)
{ cout << number*2 << endl; }

 int  main ()
 { vector<int> myVec;
    for (int i=0; i<10; i++)
       myVec.push_back (i);
       for_each (myVec.begin (), myVec.end (), MulInt);
  return 0;
}
```

a) Выведет удвоенные элементы вектора
b) Выведет элементы вектора
c) Выведет второй элемент вектора

29. Какие значения выведет данная программа?

```
#include <iostream>
#include <list>
#include <algorithm>
using namespace std;

int xform(int i)
 {  return i*10; }

int main()
{list<int> xl;
int i;
for(i=0; i<10; i++)
    xl.push_back(i);
p = transform(xl.begin(), xl.end(), xl.begin(), xform);
cout << "содержимое списка xl:";
p = xl.begin();
while(p!= xl.end())
  {    cout << *p << "  ";
       p++;}
return 0;
}
```

a) 0 1 2 3 4 5 6 7 8 9
b) 0 10 20 30 40 50 60 70 80 90
c) 0 11 12 13 14 15 16 17 18 19

30. Что вернет данная функция?

```
int gcd( int v1, int v2 )
{       while ( v1!=v2 )
   if(v1>v2)
              v1=v1-v2;
      else
              v2=v2-v1;
      return v1;
}
```

a) Максимальное из двух чисел
b) Минимальное из двух чисел
c) НОД двух чисел
d) НОК двух чисел

Проверяемая компетенция - ПК 1.3.
Проверяемые общие компетенции - ОК1-ОК9

1. Какова верная последовательность этапов программирования?
    a) компилирование, компоновка, отладка;
    b) компоновка, отладка, компилирование;
    c) отладка, компилирование, компоновка;
    d) компилирование, отладка, компоновка.

2. Какова причина синтаксических ошибок при написании программы?
   a) плохое знание языка программирования;
   b) ошибки в исходных данных;
   c) ошибки, допущенные на более ранних этапах;
   d) неправильное применение процедуры тестирования.
3. Когда можно обнаружить синтаксические ошибки в программе?
   a) при компиляции;
   b) при отладке;
   c) при тестировании;
   d) на этапе проектирования;
   e) при эксплуатации.
4. В чем заключаются ошибки компоновки программы?
   a) указано внешнее имя, но не объявлено;
   b) неправильно использовано зарезервированное слово;
   c) составлено неверное выражение;
   d) указан неверный тип переменной.
5. Могут ли проявиться ошибки в работе программы при изменении условий эксплуатации программы?
   a) да;
   b) нет.
6. Могут ли проявиться ошибки в работе программы при изменении в предметной области?
   a) да;
   b) нет.
7. Что представляет собой защитное программирование?
   a) встраивание в программу отладочных средств;
   b) создание задач защищенных от копирования;
   c) разделение доступа в программе;
   d) использование паролей;
   e) оформление авторских прав на программу.
8. Как называется вид ошибки с неправильным написанием служебных слов (операторов) при написании программы?
   a) синтаксическая;
   b) семантическая;
   c) логическая;
   d) символьная.
9. Как называется вид ошибки с неправильным использованием управляющих конструкций при написании программы?
   a) семантическая;
   b) синтаксическая;
   c) логическая;
   d) символьная.
10. Какие бывают ошибки при написании программы?
   a) синтаксические;
   b) орфографические;
   c) фонетические;
11. Как называется процедура поиска ошибки в программе, когда известно, что она есть?
   a) отладка;
   b) тестирование;
   c) компоновка;
   d) транзакция;
   e) трансляция.
12. Как называется программа для просмотра значений переменных при выполнении программы?
   a) отладчик;

b) компилятор;
c) интерпретатор;
d) трассировка;
e) тестирование.

13. Что представляет собой отладка?
a) процедура поиска ошибок, когда известно, что ошибка есть;
b) определение списка параметров;
c) правило вызова процедур (функций);
d) составление тестов для проверки работы программы.

14. Когда программист может проследить пошаговое выполнение команд программы?
a) при трассировке;
b) при тестировании;
c) при компиляции;
d) при выполнении программы;
e) при компоновке.

15. На каком этапе создания программы могут появиться синтаксические ошибки?
a) программирование;
b) проектирование;
c) анализ требований;
d) тестирование.

16. Существует ли различие между отладкой и тестированием?
a) да;
b) нет.

17. Чему нужно уделять больше времени, чтобы получить хорошую программу?
a) тестированию;
b) программированию;
c) отладке;
d) проектированию.

18. Что представляет собой трассировка программы?
a) проверка пошагового выполнения программы;
b) тестирование исходного кода;
c) отладка модуля;
d) составление блок-схемы алгоритма.

19. Что представляет собой локализация ошибки?
a) определение места возникновения ошибки;
b) определение причин ошибки;
c) обнаружение причин ошибки;
d) исправление ошибки.

20. Каково назначение отладки?
a) поиск причин существующих ошибок;
b) поиск возможных ошибок;
c) составление спецификаций;
d) разработка алгоритма.

21. Как называется инструментальное средство, не предназначенное для отладки?
a) компилятор;
b) отладчик;
c) трассировка.

22. Что такое отладка программ?
a) локализация и исправление ошибок;
b) алгоритмизация программирования;
c) компиляция и компоновка.

23. Что выполняется раньше, автономная или комплексная отладка?
a) автономная;
b) комплексная.

24. Какие действия при отладке и тестировании указывают на наличие синтаксической ошибки?
    a) Программа не компилируется
    b) Программа запускается, но работает неправильно
    c) Программа компилируется, но не запускается
25. Какие действия при тестировании указывают на наличие логической ошибки?
    a) Программа не компилируется
    b) Программа запускается, но работает неправильно
    c) Программа компилируется, но не запускается
26. Какие действия при тестировании указывают на ошибку среды выполнения?
    a) Программа не компилируется
    b) Программа запускается, но работает неправильно
    c) Программа компилируется, но не запускается
27. К каким методам при поиске ошибки относятся отладочная печать и трассировка программы?
    a) Силовые методы
    b) Метод индукции
    c) Метод дедукции
    d) Обратное движение по алгоритму
28. К какому методу при поиске ошибки относится анализ программы от частного к общему?
    a) Силовые методы
    b) Метод индукции
    c) Метод дедукции
    d) Обратное движение по алгоритму
29. К какому методу при поиске ошибки относится анализ программы от общего к частному?
    a) Силовые методы
    b) Метод индукции
    c) Метод дедукции
    d) Обратное движение по алгоритму
30. Какие действия позволяет выполнять отладчик?
    a) отслеживать, устанавливать или изменять значения переменных в процессе выполнения кода
    b) устанавливать и удалять контрольные точки
    c) выполнять трассировку программы
    d) производить оптимизацию кода
    e) компилировать исходный код

Проверяемая компетенция - ПК 1.4.
Проверяемые общие компетенции - ОК1-ОК9

1. Существует ли различие между отладкой и тестированием?
    a) да;
    b) нет.
2. Чему нужно уделять больше времени, чтобы получить хорошую программу?
    a) тестированию;
    b) программированию;
    c) отладке;
    d) проектированию.
3. Какой способ применяется для оценки надежности?
    a) тестирование;
    b) сравнение с аналогами;
    c) трассировка;
    d) оптимизация.
4. Когда можно обнаружить синтаксические ошибки?
    a) при компиляции;

b) при отладке;
c) при тестировании;
d) на этапе проектирования;
e) при эксплуатации.

5. Какова причина синтаксических ошибок при написании программы?
   a) плохое знание языка программирования;
   b) ошибки в исходных данных;
   c) ошибки, допущенные на более ранних этапах;
   d) неправильное применение процедуры тестирования.
6. Могут ли проявиться ошибки в работе программы при изменении условий эксплуатации программы?
   a) да;
   b) нет.
7. Могут ли проявиться ошибки в работе программы при изменении в предметной области?
   a) да;
   b) нет.
8. Как называется вид ошибки с неправильным написанием служебных слов (операторов) при написании программы?
   a) синтаксическая;
   b) семантическая;
   c) логическая;
   d) символьная.

9. Как называется вид ошибки с неправильным использованием управляющих конструкций при написании программы?
   a) семантическая;
   b) синтаксическая;
   c) логическая;
   d) символьная.

10. Какие ошибки существуют при написании программы?
    a) синтаксические;
    b) орфографические;
    c) фонетические;
11. Какие действия при отладке и тестировании указывают на наличие синтаксической ошибки?
    a) Программа не компилируется
    b) Программа запускается, но работает неправильно
    c) Программа компилируется, но не запускается
12. Какие действия при тестировании указывают на наличие логической ошибки?
    a) Программа не компилируется
    b) Программа запускается, но работает неправильно
    c) Программа компилируется, но не запускается
13. Какие действия при тестировании указывают на ошибку среды выполнения?
    a) Программа не компилируется
    b) Программа запускается, но работает неправильно
    c) Программа компилируется, но не запускается
14. Какое бывает тестирование программного обеспечения?
    a) автономное;
    b) инструментальное;
    c) визуальное;
    d) алгоритмическое.
15. Какое бывает тестирование программного обеспечения?

комплексное;

   a) инструментальное;
   b) визуальное;
   c) алгоритмическое.

16. Что проверяется при комплексном тестировании программного обеспечения?
    a) согласованность работы отдельных частей программы;
    b) правильность работы отдельных частей программы;
    c) быстродействие программы;
    d) эффективность программы.
17. Как называется процесс исполнения программы с целью обнаружения ошибок?
    a) тестирование;
    b) кодирование;
    c) сопровождение;
    d) проектирование.
18. Что представляет собой автономное тестирование?
    a) тестирование отдельных частей программы;
    b) инструментальное средство отладки;
    c) составление блок-схем;
    d) пошаговая проверка выполнения программы.
19. Каково назначение тестирования?
    a) обнаружение ошибок;
    b) определение степени соответствия ПО требованиям к нему
    c) повышение эффективности программы;
    d) приведение программы к структурированному виду.
20. Какой из этапов жизненного цикла программы является необязательным?
    a) оптимизация;
    b) проектирование;
    c) тестирование;
    d) программирование;
21. Как называется просчитанный вручную пример выполнения программы от исходных данных до ожидаемых результатов расчета?
    a) тест
    b) отладчик
    c) компиляция
22. Какими нужно стараться сделать тесты при их разработке?
    a) простыми
    b) сложными
23. Количество тестов определяет качество покрытия тестами программного кода?
a) да;
b) нет.
24. Как называется тестирование внутренней структуры, дизайна и кодирования программного решения, в котором код виден тестеру?
    a) Стратегией белого ящика
    b) Стратегией черного ящика
    c) Стратегией серого ящика
25. Как называется стратегия тестирования функционального поведения программы, при которой нет доступа к исходному коду приложения?
a) Стратегией белого ящика
b) Стратегией черного ящика
c) Стратегией серого ящика
26. Как называется метод тестирования программного обеспечения с неполным знанием его внутреннего устройства?
a) Стратегией белого ящика
b) Стратегией черного ящика
c) Стратегией серого ящика
27. Как называется процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом?

a) Тестированием
b) Оптимизацией
c) Отладкой
28. Как проводится ручное тестирование?
a) без использования программных средств
b) с использованием программных средств
29. Как проводится автоматизированное тестирование?
a) без использования программных средств
b) с использованием программных средств
30. Что представляет собой функциональное тестирование?
a) Тестирование программы на удобство пользовательского интерфейса
b) тестирование программы в целях проверки реализуемости функциональных требований, то есть способности в определённых условиях решать задачи, нужные пользователям
c) интенсивное использование почти готовой версии продукта с целью выявления максимального числа ошибок в его работе для их последующего устранения перед окончательным выходом продукта на рынок, к массовому потребителю

Проверяемая компетенция - ПК 1.5.
Проверяемые общие компетенции - ОК1-ОК9

1. Какой из этапов жизненного цикла программы не является обязательным?
   a) оптимизация;
   b) проектирование;
   c) тестирование;
   d) программирование;
   e) анализ требований.
2. Существует ли связь между эффективностью и оптимизацией программы?
   a) да;
   b) нет.
3. Какой способ является способом оценки надежности программы?
   a) тестирование;
   b) сравнение с аналогами;
   c) трассировка;
4. Повышает ли качество программы оптимизация кода?
   a) да;
   b) нет.
5. Как называется нахождение наилучшего варианта из множества возможных вариантов программного кода?
   a) оптимизация;
   b) тестирование;
   c) автоматизация;
   d) отладка;
   e) сопровождение.
6. Что представляет собой оптимизация программ?
   a) улучшение работы существующей программы;
   b) создание удобного интерфейса пользователя;
   c) разработка модульной конструкции программы;
   d) применение методов объектно-ориентированного программирования.
7. Какова основная цель оптимизации программы?
   a) уменьшение время выполнения или размера требуемой памяти;
   b) уменьшение размера программы;
   c) независимость модулей;
   d) качество программы, ее надежность.

8. Каков основной критерий оптимизации прграммы?
   a) эффективность использования ресурсов;
   b) структурирование алгоритма;
   c) структурирование программы.
9. Возможна ли оптимизация программ без участия программиста?
   a) да;
   b) нет.
10. Возможна ли оптимизация циклов?
   a) да;
   b) нет.
11. В чем заключается оптимизация условных выражений?
   a) в изменении порядка следования элементов выражения;
   b) в использовании простых логических выражений;
   c) в использовании сложных логических выражений;
   d) в использовании операций AND, OR и NOT.
12. В чем заключается оптимизация циклов?
   a) уменьшении количества повторений тела цикла;
   b) просмотре задачи с другой стороны;
   c) упрощение задачи за счет включения логических операций.
13. Какова суть оптимизации программы?
   a) модификация кода;
   b) отладка;
   c) повышение сложности программы;
   d) уменьшение сложности программы.
14. Каков критерий оптимизации программы?
   a) быстродействие (эффективность)
   b) надежность;
   c) мобильность.
15. Каков результат оптимизации программы?
   a) эффективность;
   b) надежность;
   c) машино-независимость;
   d) мобильность.
16. В чем заключается сущность оптимизации циклов?
   a) сокращение количества повторений тела цикла;
   b) сокращение тела цикла;
   c) представление циклов в виде блок-схем;
   d) трассировка циклов;
   e) поиск ошибок в циклах.
17. Каковы основные цели оптимизации?
   a) Уменьшение объема используемой программой оперативной памяти
   b) ускорение работы программы
   c) увеличение объема кода программы
18. Может ли случиться так, что для оптимизации работы программы часть кода переписывается на другом языке?

a) Да
b) Нет
19. Может ли оптимизация кода оказаться неэффективной?

   a) Да
   b) Нет
20. Может ли оптимизация кода оказаться нерентабельной?

a) Да
b) Нет

21. Существуют ли автоматические анализаторы кода?
a) Да
b) Нет

22. Тождественны ли понятия оптимизация кода и рефакторинг кода?
a) Да
b) Нет

23. Что в большей степени влияет на оптимальность программы?
a) Выбор алгоритма решения
b) Выбор типов данных переменных
c) Оптимизация логических выражений

24. Какое логическое выражение является более оптимальным?
a) if (5 < y && y < z)
b) if (5 < y) { if (y < z) ... }

25. Какой цикл в коде поиска наличия четного числа в массиве будет более оптимальным?
a)
```
evenNumber = false;
for (int i=0; i< count; i++)
if ((input[i] % 2) ==0)
{
        evenNumber = true;
        break;
}
```
b)
```
evenNumber = false;
for (int i=0; i< count; i++)
{
        if ((input[i] % 2) ==0)
        evenNumber = true;
}
```

26. Какие приемы можно использовать при оптимизации логического выражения при ветвлении или цикле?
a) Прекращение проверки сразу же после получения ответа
b) рекомендуется размещать ветви, вероятность выбора которых является наибольшей, ближе к началу логического выражения
c) рекомендуется размещать ветви, вероятность выбора которых является наибольшей, ближе к концу логического выражения

27. Какова операция из тождественных наболее оптимальная?
a) not a and not b
b) not (a or b)

28. Какова операция из тождественных наболее оптимальная?
a) sqrt(x) < sqrt(y)
b) x < y

29. Какой цикл будет самым оптимальным?
a)
```
for (int i = 0; i < a.Length; i++)
{
a[i] = i;
}
for (int i = 0; i < a.Length; i++)
{
b[i] = i;
}
```
b)
```
for (int i = 0; i < a.Length; i++)
{
```

```
a[i] = i;
b[i] = i;
}
```

c)
```
for (int i = 0; i < a.Length; i++)
{
a[i] = i;
for (int j = 0; j < a.Length; j++)
{
b[j] = j;
}
}
```

30. Какой код из предложенных более эффективный?
    a) a=a*16;
    b) a=a<<4;

Проверяемая компетенция - ПК 3.1.
Проверяемые общие компетенции - ОК1-ОК9

1. Какое программное обеспечение производит перевод текста программы в машинный код?
    a) компилятор
    b) отладчик
    c) редактор кода
    d) компоновщик
    e) загрузчик
2. Каково преимущество компилятора перед интерпретатором?
    a) громоздкость
    b) быстродействие
    c) малый объем
    d) удобство построчного преобразования
3. Каково преимущество интерпретатора перед компилятором?
    a) громоздкость
    b) быстродействие
    c) малый объем
4. Каков основной недостаток интерпретатора?
    a) громоздкость
    b) быстродействие
    c) малый объем
    d) низкое быстродействие
5. Что представляет собой программа в технологии структурного программирования?
    a) последовательность действий
    b) несколько взаимодействующих объектов
    c) система определений функций, описание того, что нужно вычислить
6. Что представляет собой программа в технологии объектно-ориентированного программирования?
    a) последовательность действий
    b) несколько взаимодействующих объектов
    c) система определений функций, описание того, что нужно вычислить
7. Что представляет собой программа в технологии функционального программирования?
    a) последовательность действий
    b) несколько взаимодействующих объектов
    c) система определений функций, описание того, что нужно вычислить
8. Какие принципы являются принципами структурного программирования?
    a) отказ от оператора безусловного перехода
    b) объединение данных и алгоритмов их обработки в один объект

c) любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл
d) декларирование целей программы вместо алгоритма

9. Каковы достоинства структурного программирования?
   a) Удобочитаемость программ
   b) Упрощение отладки и тестирование программ
   c) Широчайшие возможности для автоматического распараллеливания вычислений
   d) Уменьшается количество кода
10. На каком этапе жизненного цикла программы пишет техническое задание?
   a) Маркетинг и спецификация программного продукта
   b) Проектирование структуры программного продукта
   c) Документирование программного продукта
   d) Программирование
11. На каком этапе решения задачи определяется форма выдачи результатов?
   a) Постановка задачи
   b) Анализ и исследование задачи
   c) Разработка алгоритма
   d) Программирование
   e) Тестирование и отладка
12. Какие программы можно отнести к системному программному обеспечению?
   a) операционные системы;
   b) прикладные программы;
   c) игровые программы.
13. Какие программы можно отнести к системному программному обеспечению?
   a) драйверы;
   b) текстовые редакторы;
   c) электронные таблицы;
   d) графические редакторы.
14. Какие программы нельзя отнести к системному программному обеспечению?
   a) игровые программы;
   b) компиляторы языков программирования;
   c) операционные системы;
   d) системы управления базами данных
15. Какие программы можно отнести к прикладному программному обеспечению?
   a) электронные таблицы;
   b) таблицы решений;
   c) СУБД (системы управления базами данных).

16. Какие программы нельзя отнести к прикладному программному обеспечению?
   a) компиляторы и (или) интерпретаторы;
   b) текстовые и (или) графические редакторы;
   c) электронные таблицы.

17. Можно ли отнести операционную систему к программному обеспечению?
   a) да;
   b) нет.

18. Какие программы можно отнести к системному программному обеспечению?
   a) утилиты;
   b) экономические программы;
   c) статистические программы;
   d) мультимедийные программы.

19. Какой этап занимает наибольшее время в жизненном цикле программы?

a) сопровождение;
b) проектирование;
c) тестирование;
d) программирование;
e) формулировка требований.

20. Какой этап занимает наибольшее время при разработке программы?
a) тестирование;
b) сопровождение;
c) проектирование;
d) программирование;
e) формулировка требований.

21. Какой этап является первым этапом в жизненном цикле программы?
a) формулирование требований;
b) анализ требований;
c) проектирование;
d) автономное тестирование;
e) комплексное тестирование.

22. Какой этап является необязательным этапом в жизненном цикле программы?
a) оптимизация;
b) проектирование;
c) тестирование;
d) программирование;
e) анализ требований.

23. Какой этап является самым длительным в жизненном цикле программы?
a) эксплуатация;
b) изучение предметной области;
c) программирование;
d) тестирование;
e) корректировка ошибок.

24. Какой этап в жизненном цикле программы выполняется раньше других?
a) отладка;
b) оптимизация;
c) программирование;
d) тестирование.

25. Какие средства относятся к инструментальным средствам программирования?
a) компиляторы, интерпретаторы;
b) СУБД (системы управления базами данных);
c) BIOS (базовая система ввода-вывода);
d) ОС (операционные системы).

26. Какой этап в жизненном цикле программы выполняется раньше других?
a) компиляция;
b) отладка;
c) компоновка;
d) тестирование.

27. Какой этап в жизненном цикле программы выполняется раньше других?
a) проектирование;
b) программирование;
c) отладка;
d) тестирование.

28. Что относится к этапу программирования в жизненном цикле программы?
   a) написание кода программы;
   b) разработка интерфейса;
   c) работоспособность;
   d) анализ требований.
29. Какова последовательность этапов программирования?
   a) компилирование, компоновка, отладка;
   b) компоновка, отладка, компилирование;
   c) отладка, компилирование, компоновка;
   d) компилирование, отладка, компоновка.
30. На каком этапе производится выбор языка программирования?
   a) проектирование;
   b) программирование;
   c) отладка;
   d) тестирование.

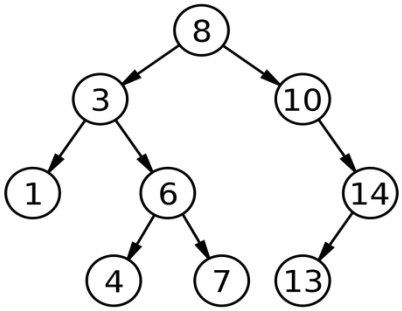
Второй блок заданий – вопросы с требуемым ответом

Проверяемая компетенция - ПК 1.1.
Проверяемые общие компетенции - ОК1-ОК9

1. Для кого предназначены утилитарные программы?
2. Для кого предназначены программные продукты?
3. Каковы этапы жизненного цикла программного продукта?
4. Каковы принципы структурного программирования?
5. Какие вы знаете современные технологии программирования?
6. Что представляет собой программа, описанная в стиле модульного программирования?
7. Чем отличаются утилитарные программы от программных продуктов?
8. Каковы основные этапы решения утилитарной задачи на ЭВМ?
9. Каковы достоинства структурного программирования?
10. Каковы основные принципы объектно-ориентированного программирования?
11. Что представляет собой инкапсуляция в технологии ООП?
12. Что представляет собой полиморфизм в технологии ООП?
13. Что представляет собой наследование в технологии ООП?
14. Каковы преимущества разделения программы на подпрограммы (модули)?
15. Что такое подпрограмма?
16. Какая функция называется рекурсивной?
17. Что такое класс в технологии ООП?
18. Из каких составляющих состоит класс в технологии ООП?
19. Чем объект отличается от класса?
20. Что представляет собой тип данных?
21. Что представляет собой стек?
22. Что представляет собой очередь?
23. Какая структура данных изображена на рисунке?

24. Какая структура данных изображена на рисунке?

25. Как называется программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных в ЭВМ?
26. Какая структура данных изображена на рисунке?



27. Какой метод доступа к элементам используется к структуре Список?
28. Какой метод доступа к элементам используется к структуре Стек?
29. Какой метод доступа к элементам используется к структуре Очередь?
30. Какая структура данных изображена на рисунке?

Проверяемая компетенция - ПК 1.2.

Проверяемые общие компетенции - ОК1-ОК9

1. В программе нужно заменить символы "+" на "-". Как будет выглядеть пропущенная команда?

```
#include <iostream>
using namespace std;

int main()
{ char s[]="lalalalalalalalalala";
  char *p=new char;
   for(p=s; *p!='\0';p++)
  if (*p=='+')
  _________
  cout<<s;
  delete p;
  return 0;}
```

2. Каким служебным словом в языках C++/C# предваряются элементы класса, которые должны быть доступны в классе и вне класса?
3. Какую спецификацию доступа имеют атрибуты класса в языках C++/C# по умолчанию?
4. Что представляет собой виртуальная функция?
5. Даны три целых числа: A, B, C. Как можно записать в виде условного выражения на языках C++/C# следующее утверждение: «Двойное неравенство A < B < C»?

6. Даны три целых числа: A, B, C. Как можно записать в виде условного выражения на языках C++/C# следующее утверждение: «Хотя бы одно из чисел A, B, C положительное»?
7. Какие существуют спецификации доступа к членам класса на языках C++/C#?
8. Дайте определение конструктора. Каково назначение конструктора?
9. Сколько конструкторов может быть в классе?
10. Из чего состоит описание пользовательской функции на языках C++/C#?
11. В чем состоит механизм перегрузки операций?
12. Какими способами можно заполнить массив данными?
13. Какие категории простых типов данных в языках программировании высокого уровня вы можете назвать?
14. Какие требования к идентификаторам существуют в большинстве языков программирования?
15. В чем состоит разница между операторами break и continue в языках C++/C#?
16. Что представляет собой тип данных?
17. В чем разница между операциями префиксного и постфиксного инкремента (x++ и ++x) в языках C++/C#?
18. Как может выглядеть рекурсивная функция вычисления факториала целого числа на языках C++/C#?
19. Что представляет собой тип данных указатель?
20. Что выполнит данный фрагмент программы?

```
int main()
{
int a[5]={0,1,2,3,4};
int *p;
int sum=0;
for(p=a;p<a+5;p++)
sum=sum+*p;
cout<<“sum = "<<sum<<endl;
return 0;
}
```

21. Что выполнит данная программа?

```
#include <iostream>
using namespace std;
int length(char *s)
{int i;
  for(i=0; *s!='\0';s++)
    i++;
 return(i);
 }
int main()
{ char s[]="lalalalalalalalalala";
  cout<<length(s);
  return 0;}
```

22. Данная команда открытия файла ifstream input_file("FILENAME.DAT"); на языке C++ откроет его как текстовый или двоичный?
23. В данном фрагменте для чего используется функция fail?

```
ifstream input_file("FILENAME.DAT");
if (input_file.fail())
{
  cerr << "Ошибка открытия FILENAME.EXT" << endl;
  exit(1);
}
```

24. Что выполнит команда input_file.close() на языке C++**?**
25. Что выполнит данная программа?

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
  ifstream in_file("f.txt");
  char x;
  if (in_file.fail())
  {
    cout << "Error";
    exit(1);
  }
int count1 = 0;
  int count2 = 0;
  while (in_file >> x)
  {
    if (x == '{') count1++;
    if (x == '}') count2++;
  }
  in_file.close();
  if (count1 == count2)
    cout << "Program good!";
}
```

26. Как будут называться следующие функции в одной программе?

```
int add_values(int a,int b)
{   return(a + b);
}
 int add_values (int a, int b, int c)
{
   return(a + b + c);
}
```

27. Что выполнит следующая программа?

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;
 bool myfunction (int i,int j)
 { return (i>j); }
```

```
int main ()
 { int myints[] = {32,71,12,45,26,80,53,32};
vector<int> myvector (myints, myints+8);
 sort (myvector.begin(), myvector.end(), myfunction);
 for (vector<int>::iterator it=myvector.begin(); it!=myvector.end(); it++)
 cout << *it << ' ';
 return 0; }
```

28. Что выполнит следующая программа?

```
#include <iostream>
#include <list>
#include <algorithm>

 using namespace std;

int xform(int i)            // Простая функция преобразования

 {  return i*i; }
 int main()

{   list<int> xl;

    int i;

    for(i=0; i<10; i++)
     xl.push_back(i);
    p = transform(xl.begin(), xl.end(), xl.begin(), xform);
    return 0;

}
```

29. Какая форма ветвления представлена следующим оператором?

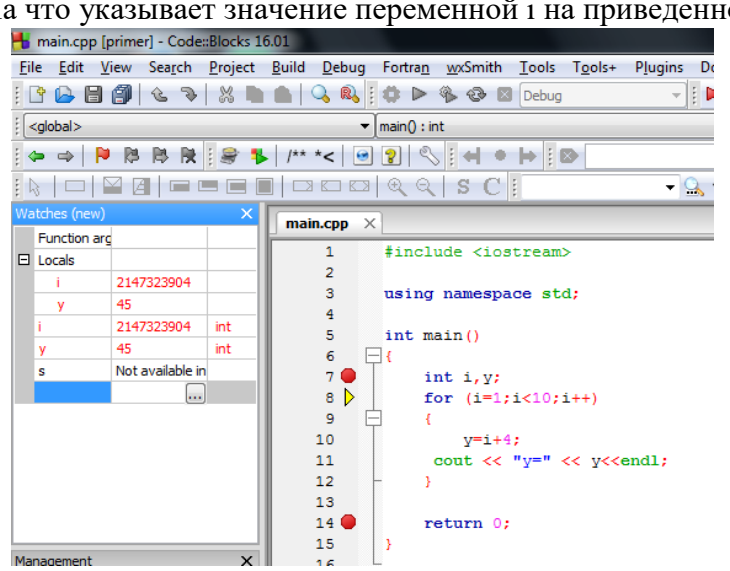
```
if (Условие)
{
 БлокОпераций1;
}
```

30. Какая форма ветвления представлена следующим оператором?

```
if (Условие)
{
 БлокОпераций1;
}
else
{
 БлокОпераций2;
}
```

Проверяемая компетенция - ПК 1.3.
Проверяемые общие компетенции - ОК1-ОК9

1. На что указывает значение переменной i на приведенном скриншоте отладки?



2. Как называется просчитанный вручную пример выполнения программы от исходных данных до ожидаемых результатов расчета?
3. Какие ключевые слова C++ используются для обработки исключений?
4. Что обозначается ключевым словом catch в коде программы?
5. Что обозначается ключевым словом try в коде программы?
6. Что представляет собой отладка программы?
7. Какие средства облегчают процесс отладки?
8. Что представляет собой аналитический способ обнаружения ошибки?
9. Что представляет собой экспериментальный способ обнаружения ошибки?
10. Что представляет собой отладочная печать?
11. Программа компилируется и работает, но выдает неправильный результат. Сама логика, на которой базируется вся программа, является ущербной. К какому типу ошибок относится подобная ошибка?
12. Иногда, синтаксис исходного кода может быть безупречным, но ошибка все же может произойти. Это может быть связано с проблемами в самом компиляторе. К какому типу ошибок относится подобная ошибка?
13. Программный код успешно скомпилирован, и исполняемый файл был создан. Ошибки при выполнении программы могут возникнуть в результате аварии или нехватки ресурсов носителя. К какому типу ошибок относится подобная ошибка?
14. Какие действия можно отнести к силовым методам отладки?
15. Как исправить ошибку в функции вычисления среднего арифметического? Напишите верный код.

```
float average(float a, float b)
{
  return a + b / 2;
}
```

16. Что представляет собой исключительная ситуация?
17. Если при выполнении операторов контролируемого блока программы исключений не возникло, используются ли функции-обработчики?
18. Если в контролируемом блоке при выполнении программы формируется исключение, что произойдет?
19. Как можно исправить ошибку в следующем коде программы? Напишите правильный код.

```
int arr[10];
for (i = 1; i <= 11; i++)
    arr[i] = 0;
```

20. Как можно исправить ошибку в следующем коде программы? Напишите правильный код.

```
for (int i=0; i < n;)
    i++;
```

```
        sum =sum + i;
```

21. Существуют ли автоматизированные средства отладки, позволяющие найти и исправить ошибки без участия человека, не беря в расчет модульные тесты?
22. Что представляет собой трассировка программы?
23. Что представляет собой точка останова?
24. Как называется программа пошагового выполнения для просмотра значений переменных при выполнении программы?
25. В чем разница между отладкой и тестированием?
26. Как можно исправить ошибку в следующем коде программы? Напишите правильный код.

```
int sum;
for (int i=0; i < n; i++)
    sum =sum + a[i];
```

27. Как можно исправить ошибку в следующем коде программы? Напишите правильный код.

```
int sum=0;
for (int i; i < n; i++)
    sum =sum + a[i];
```

28. Как можно исправить ошибку в следующем коде программы? Напишите правильный код.

```
int product=0;
for (int i=0; i < n; i++)
    product = product * a[i];
```

29. Какую ошибку можно получить, применяя данную функцию?

```
int average(int a, int b)
{
  return (a + b) / 2;}
```

30. Какие результаты выведет данная программа?

```
class Otr {};

float kor(float n)
{
    if (n < 0) throw Otr();
    double b = sqrt(n);
    return b;
 }

int main()
{   float x,y;
     x= -25;
     try
     {
          y= kor(x);
     }
     catch (Otr)
    {
          cerr << "Negative digit"; y = kor(-x); \
     }
  cout << "\nResult: " << y;
}
```

Проверяемая компетенция - ПК 1.4.
Проверяемые общие компетенции - ОК1-ОК9

1. Как называется просчитанный вручную пример выполнения программы от исходных данных до ожидаемых результатов расчета?

2. Программа компилируется и работает, но выдает неправильный результат. Сама логика, на которой базируется вся программа, является ущербной. К какому типу ошибок относится подобная ошибка?
3. Иногда, синтаксис исходного кода может быть безупречным, но ошибка все же может произойти. Это может быть связано с проблемами в самом компиляторе. К какому типу ошибок относится подобная ошибка?
4. Программный код успешно скомпилирован, и исполняемый файл был создан. Ошибки при выполнении программы могут возникнуть в результате аварии или нехватки ресурсов носителя. К какому типу ошибок относится подобная ошибка?
5. В чем разница между отладкой и тестированием?
6. Какие стратегии тестирования Вы знаете?
7. Каким образом составляются тестовые наборы для ручного тестирования?
8. Как называется интенсивное использование почти готовой версии продукта с целью выявления максимального числа ошибок в его работе для их последующего устранения перед окончательным выходом продукта к массовому потребителю?
9. Что такое функциональное тестирование?
10. Что такое ручное тестирование?
11. Что такое автоматизированное тестирование?
12. При разработке тестов какими нужно стараться их сделать, простыми или сложными?
13. Как называется стратегия тестирования функционального поведения программы, при которой нет доступа к исходному коду приложения?
14. Как называется тестирование внутренней структуры, дизайна и кодирования программного решения, в котором код виден тестеру?
15. Как называется тестирование отдельных частей программы?
    Каково назначение тестирования?
16. Что такое дефект, или баг?
17. Могут ли проявиться ошибки в работе программы при изменении условий эксплуатации программы?
18. Могут ли проявиться ошибки в работе программы при изменении в предметной области?
19. Существует ли различие между отладкой и тестированием?
20. Как называется тестирование отдельного модуля программы?
21. Как называется тестирование группы модулей программы?
22. Исправьте ошибку в функции вычисления среднего арифметического

```
float average(float a, float b)
{
return a + b / 2;
}
```

23. Имеется код:

```
Int a,b;
Cin>>a>>b;
If (a%b==0)
{
Cout<<a<<”  кратно ”<<b;
}
```

Нужен ли в тест-кейсах следующий тест?

3 0

24. Имеется код:

```
Int a;
Cin>>a;
If (a%2==0)
{
Cout<<a<<”  четное ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?
25. Имеется код:

```
Int a;
Cin>>a;
If (a%2==1)
{
Cout<<a<<”  нечетное ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?
26. Имеется код:

```
Int a;
Cin>>a;
If (a>=0)
{
Cout<<a<<”  неотрицательное ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?
27. Имеется код:

```
Int a;
Cin>>a;
If (a<0)
{
Cout<<a<<”  отрицательное ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?
28. Имеется код:

```
Int a;
Cin>>a;
If (a>0)
{
Cout<<a<<”  положительное ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?
29. Имеется код:

```
Int a;
Cin>>a;
If (a==0)
{
Cout<<a<<”  нулевое ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?
30. Имеется код:

```
Int a;
Cin>>a;
If (a!=0)
{
Cout<<a<<”  ненулевое ”;
}
```

Какое минимальное количество тестов нужно сделать, чтобы проверить конструкцию ветвления?

Проверяемая компетенция - ПК 1.5.

Проверяемые общие компетенции - ОК1-ОК9

1. Как определить «узкие» места программы, требующие оптимизации?
2. Что представляет собой оптимизация кода?
3. Каковы цели оптимизации?

4. Тождественны ли понятия оптимизация кода и рефакторинг кода?
5. Чем отличается оптимизация кода от рефакторинга?
6. Существуют ли автоматизированные средства оптимизации кода?
7. Могут ли автоматизированные средства оптимизации кода кардинально изменить первоначальный алгоритм?
8. Может ли оптимизация кода оказаться неэффективной?
9. Может ли оптимизация кода оказаться нерентабельной?
10. Какова сущность оптимизации циклов?
11. Является ли оптимизированным следующий код поиска наличия четного элемента в массиве?

```
evenNumber = false;
for (int i=0; i< count; i++)
if ((input[i] % 2) ==0)
{
        evenNumber = true;
        break;
}
```

12. Возможна ли оптимизация программ без участия программиста?
13. Возможна ли оптимизация циклов?
14. Является ли оптимизация обязательным этапом жизненного цикла программы?
15. Какая из двух записей условия будет наиболее оптимальной и почему?

```
not a and not b
not (a or b)
```

16. Может ли операция break помочь оптимизировать цикл?
17. Дан алгоритм пузырьковой сортировки. Есть ли в нем приемы оптимизации кода?

```
ЦИКЛ ДЛЯ J=1 ДО N-1 ШАГ 1
 F=0
 ЦИКЛ ДЛЯ I=0 ДО N-1-J ШАГ 1
    ЕСЛИ A[I] > A[I+1] ТО
      ОБМЕН A[I],A[I+1]
      F=1
 ЕСЛИ F=0 ТО ВЫХОД ИЗ ЦИКЛА
```

18. Какая операция выполнится быстрее, умножение на 2 или битовый сдвиг влево?
19. В целях экономии памяти какие переменные следует предпочитать в программе, глобальные или локальные?
20. Влияют ли характеристики вычислительной машины (например, количество и тактовая частота процессорных ядер, размер кэша, пропускная способность системной шины, объём оперативной памяти) на оптимизацию программы?
21. Как можно оптимизировать код, если два соседних цикла повторяются одно и то же число раз и не влияют друг на друга?
22. Каковы минусы оптимизации?
23. Может ли оптимизация привести к появлению ошибок?
24. Нужно ли оптимизировать весь код программы?
25. При написании программ в целях оптимизации предпочтительнее использовать стандартные библиотечные функции или писать вместо них свои?
26. Данный код заменяет прописные буквы на строчные:

```
void lower(char *s) {
   for (int i = 0; i < strlen(s); i++)
      if (s[i] >= 'A' && s[i] <= 'Z')
         s[i] -= ('A' - 'a');
}
```

Что существенно замедляет работу этой функции?

27. Какая операция выполнится быстрее, деление на 2 или битовый сдвиг вправо?
28. Дан алгоритм пузырьковой сортировки. Есть ли в нем приемы оптимизации кода?

```
ЦИКЛ ДЛЯ J=1 ДО N-1 ШАГ 1
```

```
ЦИКЛ ДЛЯ I=0 ДО N-1 ШАГ 1
   ЕСЛИ A[I] > A[I+1] ТО
    ОБМЕН A[I],A[I+1]
```

29. Является ли оптимизированным следующий код поиска наличия четного элемента в массиве?

```
evenNumber = false;
for (int i=0; i< count; i++)
if ((input[i] % 2) ==0)
{
        evenNumber = true;
}
```

30. Арифметические операции выполняется быстрее с целыми или вещественными числами?

Проверяемая компетенция - ПК 3.1.
Проверяемые общие компетенции - ОК1-ОК9

1. На какие классы делятся программные продукты по сфере использования?
2. Как можно охарактеризовать понятие «средства для создания приложений»?
3. Как можно охарактеризовать понятие «язык программирования»?
4. Как называются в языке программирования правила построения допустимых сообщений?
5. Как называются в языке программирования смысл и логика синтаксических конструкций?
6. Для кого предназначены утилитарные программы?
7. Для кого предназначены программные продукты?
8. Что представляет собой компилятор?
9. Что представляет собой интерпретатор?
10. Чем отличается компилятор от интерпретатора?
11. В чем состоит сходство компилятора и интерпретатора?
12. В чем состоит преимущество компилятора перед интерпретатором?
13. В чем состоит недостаток компилятора в сравнении с интерпретатором?
14. Какой процесс выполнится раньше, компиляция или компоновка?
15. Куда загрузчик размещает исполняемую программу?
16. Каковы этапы жизненного цикла программного продукта?
17. Каковы принципы структурного программирования?
18. Какие Вы знаете современные технологии программирования?
19. Что представляет собой программа, описанная в стиле модульного программирования?
20. Чем отличаются утилитарные программы от программных продуктов?
21. Каковы основные характеристики программного продукта?
22. Какой процесс выполнится раньше Редактирование кода или Препроцессорная обработка?
23. Каковы основные этапы решения утилитарной задачи на ЭВМ?
24. Каковы достоинства структурного программирования?
25. Каковы основные принципы объектно-ориентированного программирования?
26. Что представляет собой принцип инкапсуляции?
27. Что представляет собой принцип полиморфизма?
28. Что представляет собой принцип наследования?
29. Каковы преимущества разделения программы на подпрограммы (модули)?
30. Что такое подпрограмма?

Составила Мохнач О.А.