Тест по курсу C++

Теоретические вопросы

1. Каким ключевым словом обозначаются встраиваемые функции? inline

Ваш ответ:

2. Каким будет значение переменной `c` после выполнения следующего кода?

```
int a, b = a = 0, c;
if (b != a++)
    c = 1;
else if (b == --a)
    c = 2;
```

Варианты ответа:

- 0
- 1
- 2
- код не скомпилируется.

Ваш ответ:

3. Пусть `a = -5`. Сколько раз выполнится тело цикла

```
while (a<0) a++;
```

Варианты ответа:

- 0
- 4
- 5
- 6

Ваш ответ:

4. Какие ключевые слова входят в состав инструкции "если":

Выберите возможные варианты:

- `Else`
- `then`
- `if`
- `elseif`
- `unless`

Ваш ответ:

5. Что из перечисленного является объявлением указателя в C++:

- `int* a;`
- `int &a;`
- `int a&;`

- `int* &a;`

Ваш ответ:

6. Дана функция:

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

Как правильно вызвать эту функцию?

Выберите возможные варианты:

- `int sum(int a = 7, int b = 8);`
- `sum(int 7, int 8);`
- `sum(7, 8);`
- `sum() : 7, 8;`
- `sum(int a = 7, int b = 8);`

Ваш ответ:

7. Выберите правильные утверждения о конструкторе класса:

- Конструктор объявляется в точности так же, как и обычный метод класса.
- Конструктор не возвращает значения.
- Конструктор может иметь любое имя.
- Конструктор имеет то же имя, что и класс.
- Имя конструктора начинается с символа ~.

Ваш ответ:

8. Содержит ли этот код ошибки?

```
class A {
  static void Foo();
};
void A::Foo() { }
```

Варианты ответа:

- да
- нет

Ваш ответ:

9. Какие виды комментариев есть в C++?

Выберите возможные варианты:

- `# comment`
- `// comment`
- `; comment`
- `<!-- comment -->`
- `/* comment */`

Ваш ответ:

10. Что означает конструкция

```
throw()
```

в объявлении функции? (например `void f() throw();`)

Варианты ответа:

- Такое объявление указывает, что функция не должна генерировать исключения.
- Такое объявление указывает, что функция может сгенерировать любое исключение.
- Такое объявление не корректно, произойдет ошибка времени компиляции.
- Такое объявление указывает, что функция может сгенерировать исключения только из стандартной библиотеки или же исключения такого же типа как и параметры функции.

Ваш ответ:

11. В каком порядке инициализируются поля в классе?

Варианты ответа:

- Порядок инициализации не гарантируется
- В порядке перечисления инициализаторов в списке инициализации конструктора
- В порядке их объявления

Ваш ответ:

12. Что верно о следующем коде:

```
int main(int argc, char* argv[])
{
  int a[3] = { 1, 2, 3 };
  int b[2] = { 1, 2 };
  a = b;
  return 0;
}
```

Варианты ответа:

- Код скомпилируется и успешно выполнится
- Код скомпилируется, но возникнет ошибка времени выполнения
- Код не скомпилируется

Ваш ответ:

13. Сколько раз выполнится цикл:

```
for (int i = 0; i <= 5; i += 3);
```

Варианты ответа:

- 5 раз,

- 6 раз
- это бесконечный цикл,
- ни одного раза,
- 2 раза,
- 3 раза

Ваш ответ:

14. Как правильно объявить массив?

Варианты ответа:

- `int array[];`
- `int array[5];`
- `int array[] = new array[5];`
- `int array[] = new int[];`

Ваш ответ:

15. Что произойдет, если объявить следующие функции:

```
int Square (int width, int length = 3);
int Square (int size);
```

и вызвать функцию

```
int s = Square (10);
```

Варианты ответа:

- Код скомпилируется и успешно выполнится
- Код скомпилируется, но возникнет ошибка времени выполнения
- Код не скомпилируется

Ваш ответ:

16. Какой будет вывод в консоль и почему?

```
#include <iostream>
#include <string>

void print(int v)
{
    std::cout << "int:" << v << std::endl;
}

void print(bool v)
{
    std::cout << "bool:" << v << std::endl;
}

void print(std::string v)
{
    std::cout << "std::string:" << v << std::endl;
}

int main()
{
```

```
    print(1);
    print(true);
    print("Hello world");
}
```

Ваш ответ:

int:1

bool:1

std::string:Hello world

17. Для каких целей применяется ключевое слово const?

Ваш ответ: указывает, что значение является константой и сообщает о том, что значение не может быть изменено после инициализации

18. Как защитить объект от копирования?

Ваш ответ:

использование = delete:

Практические вопросы

1. Создайте класс “A”, инкапсулирующий динамический массив. Введите в класс необходимые на Ваш взгляд данные и методы, чтобы приведенный ниже код выполнялся:

```
{
    A a1;
    A a2(10); //10 - размер массива
    A a3 = a2;
    a1 = a3;
    a2 = A(20);
    const A a4(5);
    for(int i=0; i<a2.size(); i++)
    {
    cout<<a4[i];
    }
}
```

```
class A {
private:
    int* arr;
    int arr_size;

public:
```

```
    A() : arr(nullptr), arr_size(0) {}

    A(int size) : arr(new int[size]()), arr_size(size) {}

    A(const A& other) : arr(new int[other.arr_size]()), arr_size(other.arr_size) {
        copy(other.arr, other.arr + other.arr_size, arr);
    }

    A& operator=(const A& other) {
        if (this == &other) return *this;
        delete[] arr;
        arr_size = other.arr_size;
        arr = new int[arr_size]();
        copy(other.arr, other.arr + other.arr_size, arr);
        return *this;
    }

    ~A() {
        delete[] arr;
    }

    int size() const {
        return arr_size;
    }

    int& operator[](int index) {
        if (index < 0 || index >= arr_size) {
            throw out_of_range("индекс за границами массива");
        }
        return arr[index];
    }

    const int& operator[](int index) const {
        if (index < 0 || index >= arr_size) {
            throw out_of_range("индекс за границами массива");
        }
        return arr[index];
    }
};
```

2. Добавьте в программу Eng_float модуля 12 в класс Distance перегруженную операцию присваивания.

```
        Distance& operator=(const Distance& other)
        {
                if (this == &other)
                        return *this;
                this->feet = other.feet;
                this->inches = other.inches;

                return *this;
```

```
    }
};
```

3. Дан класс Point:

```
class Point{
int x,y;
…
};
```

Обеспечьте выполнение:

```
{
     Point pt1(1,1) pt2(2,2), pt3;
     pt3 = pt1 + pt2;
     pt2 +=pt1;
     pt3 = pt1 + 5;
     …
}
```

```
#include <iostream>

using namespace std;

class Point {
private:
    int x, y;
public:
    Point() : x(0), y(0) {}
    Point(int x, int y) : x(x), y(y) {}
    Point operator+(const Point& pt) const {
        return Point(x + pt.x, y + pt.y);
    }
    Point& operator+=(const Point& pt) {
        x += pt.x;
        y += pt.y;
        return *this;
    }
    Point operator+(int val) const {
        return Point(x + val, y + val);
    }
    void show() const {
        cout << "(" << x << ", " << y << ")";
```

```
    }
};

int main() {
    Point pt1(0, 5), pt2(3, 4), pt3;

    pt3 = pt1 + pt2;
    cout << "pt3 = pt1 + pt2 ";
    pt3.show();
    cout << endl;

    pt2 += pt1;
    cout << "pt2 += pt1 ";
    pt2.show();
    cout << endl;

    pt3 = pt1 + 5;
    cout << "pt3 = pt1 + 5 ";
    pt3.show();
    cout << endl;

    return 0;
}
```

4. Напишите класс, который реализует функционал стека. Класс **Stack** должен иметь:

- Открытый целочисленный фиксированный массив длиной 10.
- Открытое целочисленное значение для отслеживания длины стека.
- Открытый метод с именем reset(), который будет сбрасывать длину и все значения элементов на 0.
- Открытый метод с именем push(), который будет добавлять значение в стек. Метод push() должен возвращать значение false, если массив уже заполнен, в противном случае — true.
- Открытый метод с именем pop() для возврата значений из стека. Если в стеке нет значений, то должно выводиться предупреждающее сообщение.
- Открытый метод с именем print(), который будет выводить все значения стека.

Следующий код функции main():

```
int main()
```

```
{
Stack stack;
stack.reset();
stack.print();
stack.push(3);
stack.push(7);
stack.push(5);
stack.print();
stack.pop();
stack.print();
stack.pop();
stack.pop();
stack.print();
return 0;
}
```

Должен производить следующий результат:

```
( )
( 3 7 5 )
( 3 7 )
( )
```

```
#include <iostream>

using namespace std;

class Stack {
public:
   int arr[10];
   int length;

   Stack() : length(0) {
      reset();
   }

   void reset() {
      length = 0;
      for (int i = 0; i < 10; ++i)
         arr[i] = 0;
   }
```

```
    bool push(int value) {
        if (length >= 10) {
            return false;
        }
        arr[length++] = value;
        return true;
    }

    int pop() {
        return arr[--length];
    }

    void print() const {
        cout << "( ";
        for (int i = 0; i < length; ++i)
            cout << arr[i] << " ";
        cout << ")" << endl;
    }
};

int main() {
    Stack stack;
    stack.reset();
    stack.print();
    stack.push(3);
    stack.push(7);
    stack.push(5);
    stack.print();
    stack.pop();
    stack.print();
    stack.pop();
    stack.pop();
    stack.print();

    return 0;
}
```