Тест по C++ для студентов ШАД. 9 февраля 2011 г.

Выберите правильный вариант ответа. Везде подразумевается, что платформа 32-битная. Предупреждения компилятора не считаются ошибками. Считается, что все программы начинаются со следующих строк:

```
#include <iostream>
#include <vector>
#include <map>
#include <set>
#include <string>
using namespace std;
```

Впишите фамилию и имя:

_____

Вопрос 1. Что напечатает программа?

```
int main() {
  const char* a = "abcd";
  const char b[] = "abcd";
  cout << sizeof a << ' ' << sizeof b << endl;
}
```

(1) `4 4`;
(2) `4 5`;
(3) `5 5`;
(4) программа не скомпилируется.

Вопрос 2. Что произойдет с программой?

```
int main() {
  int i = 1;
  switch (i) {
    case 0:
      cout << 0 << endl;  break;
    otherwise:
      cout << 1 << endl;
  }
}
```

(1) напечатает `1`;
(2) напечатает `0` и `1`;
(3) ничего не напечатает;
(4) не скомпилируется.

Вопрос 3. Что напечатает программа?

```
int first() {
  int i = 0;
  return i++;
}
int second() {
  static int i = 0;
  return i++;
}
int main() {
  for (int k = 0; k < 3; ++k)
    cout << first() << " ";
  for (int k = 0; k < 3; ++k)
    cout << second() << " ";
}
```

(1) `0 0 0 0 0 0`;
(2) `0 0 0 0 1 2`;
(3) `0 0 0 1 2 3`;
(4) `1 1 1 1 1 1`;
(5) `1 1 1 1 2 3`.

Вопрос 4. Что делает функция стандартной библиотеки `unique`?

(1) удаляет повторяющиеся элементы из последовательности;
(2) переупорядочивает элементы, сдвигая повторяющиеся элементы в конец;
(3) переупорядочивает элементы, оставляя для каждой последовательной серии дубликатов только первый элемент;
(4) считает количество уникальных элементов в последовательности;
(5) такой функции нет в стандартной библиотеке.

Вопрос 5. Что напечатает программа?

```
int main() {
  vector<int> v;
  for (size_t i = 0; i < 10; ++i)
    v.push_back(i);
  int* i1 = &v[0] + 3;
  int* i2 = &v[0] + 9;
  for (size_t i = 0; i < 10; ++i)
    v.push_back(i);
  int* i3 = &v[0] + 11;
  int* i4 = &v[0] + 15;
  cout << *i1 + *i2 + *i3 + *i4 << endl;
}
```

(1) `14`;
(2) `18`;
(3) ничего, программа не скомпилируется;
(4) поведение программы не определено.

Вопрос 6. Чем отличаются перечисленные способы найти элемент в множестве?

```
set<int> s;
// fill set with elems...
set<int>::iterator it;
// it = find(s.begin(), s.end(), 5); // a
// it = s.find(5);                   // b
```

(1) a асимптотически быстрее b;
(2) b асимптотически быстрее a;
(3) a нельзя использовать с `set`;
(4) ничем не отличаются.

Вопрос 7. Чем отличаются перечисленные способы найти элемент в ассоциативном массиве?

```
map<int, string> m;
// fill map with elems...
map<int, string>::iterator it;
// it = find(m.begin(), m.end(), 5); // a
// it = m.find(5);                   // b
```

(1) a асимптотически быстрее b;
(2) b асимптотически быстрее a;
(3) a нельзя использовать с `map`;
(4) ни a, ни b нельзя использовать с `map`;
(5) ничем не отличаются.

Вопрос 8. Есть ли ошибка в следующей программе? Если да, то исправьте.

```
int* a = new int[10];
for (size_t i = 0; i < 10; ++i)
  *(a + i) = i;
delete a;
```

Впишите ответ:

_____

Вопрос 9. Что напечатает программа?

```
struct Foo {
  int foo;
  Foo(int& f): foo(f) {
    foo++;
  }
  void print() {
    cout << foo++;
  }
};
int main() {
  Foo f(1);
  f.print();
}
```

(1) 1;
(2) 2;
(3) 3;
(4) ничего, программа не скомпилируется.

Вопрос 10. Что напечатает программа?

```
struct Base {
  Base() { cout << "Base()"; }
  virtual ~Base() { cout << "~Base()"; }
};
class Derived : public Base {
public:
  Derived() { cout << "Derived()"; }
  ~Derived() { cout << "~Derived()"; }
};
int main() {
  Base* d = new Derived;
  delete d;
}
```

(1) `Base() Derived() ~Base() ~Derived();`
(2) `Base() Derived() ~Derived() ~Base();`
(3) `Base() ~Base() Derived() ~Derived();`
(4) Поведение программы не определено.

Вопрос 11. Что напечатает программа?

```
class Rectangle {
public:
  const int area;
  const int width;
  const int height;
public:
  Rectangle(int i_width, int i_height):
    width(i_width), height(i_height), area(width*height) {}
};
int main() {
  Rectangle sample(10, 5);
  cout << sample.area << endl;
}
```

(1) 50;
(2) 0;
(3) программа не скомпилируется;
(4) поведение программы не определено.

Вопрос 12. Как будет работать программа?

```
class DataHolder {
private:
  int data;
public:
  DataHolder(): data(5) {}
  DataHolder(const DataHolder& old) { *this = old; }
  DataHolder operator = (DataHolder oldDataHolder) {
    data = oldDataHolder.data;
    return *this;
  }
};
int main() {
  DataHolder var1;
  DataHolder var2(var1);
}
```

(1) программа не скомпилируется;
(2) программа корректно завершит работу;
(3) произойдет переполнение стека;

Вопрос 13. Что напечатает программа?

```
struct C {
  int data;
  C(int i = 0) : data(i) {
    cout << "C(" << data << ")" << " ";
  }
  C& operator = (const C& other) {
    cout << "operator =" << " ";
  }
};
int main() {
  map<int, C> m;
  m[0] = C(10);
}
```

(1) `C(10) C(0) operator =`;
(2) `C(10) operator =`;
(3) `C(0) C(10)`;
(4) `C(10)`.

Вопрос 14. Какие строчки являются корректными?

```
struct C {
  C() {}
  void first() {}
  void second() const {}
};
int main() {
  C c1;
  const C c2;
  // c1.first();  // a
  // c1.second(); // b
  // c2.first();  // c
  // c2.second(); // d
}
```

(1) все;
(2) все, кроме b;
(3) все, кроме c;
(4) все, кроме b и c.

Вопрос 15. Что напечатает программа?

```
struct Num {
  int value;
  Num operator ++ (int) {
    Num copy(*this);
    value++;
    return copy;
  }
  Num& operator ++ () {
    ++value;
    return *this;
  }
};
int main() {
  Num i;
  i.value = 1; ++++i; cout << i.value << " ";
  i.value = 1; i++++; cout << i.value << endl;
}
```

(1) `1 1`;
(2) `3 1`;
(3) `3 2`;
(4) `3 3`;
(5) программа не скомпилируется.

Вопрос 16. Укажите области видимости внутри класса `Derived`:

```
class Base {
public:
  int d1;
protected:
  int d2;
private:
  double d3;
};
class Derived: protected Base {
private:
  float e1;
};
```

(1) `d1`, `d2`, `d3` – `protected`; `e1` – `private`;
(2) `d1` – `public`; `d2` – `protected`; `d3`, `e1` – `private`.
(3) `d1`, `d2` – `protected`; `d3`, `e1` – `private`.
(4) среди вышеперечисленных вариантов нет правильного ответа.

Вопрос 17. Что напечатает программа?

```
class Base {
public:
  void f() { cout << "Base::f" << endl; }
  virtual ~Base() {}
};
class Derived: public Base {
public:
  void f() { cout << "Derived::f" << endl; }
};
int main() {
  Base* b = new Derived;
  b->f();
  delete b;
}
```

(1) `Base::f`;
(2) `Derived::f`;
(3) ничего, программа не скомпилируется;
(4) поведение программы не определено.

Вопрос 18. Что напечатает программа?

```
template <class T> T f() {
  static int i = 0;
  return ++i;
}
int main() {
  cout << f<int>() << " ";
  cout << f<short>() << " ";
  cout << f<int>() << " ";
}
```

(1) `1 1 1`;
(2) `1 2 3`;
(3) `1 1 2`;
(4) ничего, программа не скомпилируется;
(5) результат не определен.

Вопрос 19. Что напечатает программа?

```
template<typename T>
struct Bar {
  enum { result = 1 };
};
template<typename T>
struct Bar<T*> {
  enum { result = 1 + Bar<T>::result };
};
int main() {
  cout << Bar<int***>::result << endl;
}
```

(1) 1;
(2) 3;
(3) 4;
(4) компилятор зациклится.

Вопрос 20. Что напечатает программа?

```
template<int N, int M = 0>
class Foo: public Foo<N/2, M+1> {
};
template<int M>
class Foo<0, M> {
public:
  static const int Result = M;
};
int main() {
  cout << Foo<11>::Result - Foo<3,2>::Result << endl;
}
```

(1) 0;
(2) 1;
(3) поведение программы не определено;
(4) ничего, программа не скомпилируется.