МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

Национальный исследовательский ядерный университет «МИФИ»

Факультет Кибернетики и Информационной безопасности

КАФЕДРА КИБЕРНЕТИКИ (№ 22)

Направление подготовки 09.03.04 Программная инженерия

Объектно-ориентированное программирование

Лабораторная работа 1.
Изучение базовых конструкций языка C++

Группа _____ M20-504

Студент _____ Авраменко А. Д.

Преподователь _____ Шапкин П.А.

Москва 2021

Задание 1

Задание 1. *Написать программы, использующие функции потокового ввода-вывода:*

- *с полным указанием пространства имен std;*
- *с сокращенным указанием пространства имен (использовать using);*
- *вызывающую ошибку компиляции из-за отсутствия указания пространства имен.*

Первые две части задания (1 и 2) компилируются и исполняются без ошибок, как ожидается. Третья часть (3) не компилируется из-за недекаларированных переменных (4), как ожидается.

```
#include <iostream>
#include <cstring>



int main()
{
  std::string tmp;
  std::cin >> tmp;
  std::cout << tmp << std::endl;
}
```

Листинг 1 – с полным указанием пространства имен std

```
#include <iostream>
#include <cstring>



int main()
{
  using namespace std;
  string tmp;
  cin >> tmp;
  cout << tmp << endl;
}
```

Листинг 2 – с сокращенным указанием пространства имен (использовать using)

```
#include <iostream>
#include <cstring>



int main()
{
  string tmp;

  cin >> tmp;
  cout << tmp << endl;
}
```

Листинг 3 – вызывающую ошибку компиляции из-за отсутствия указания пространства имен

```
1_3lab.cpp: In function ‘int main()’:
1_3lab.cpp:9:2: error: ‘’string was not declared in this scope
  string tmp;
  ^~~~~~
1_3lab.cpp:9:2: note: suggested alternative:
In file included from /usr/include/c++/8/iosfwd:39,
                 from /usr/include/c++/8/ios:38,
                 from /usr/include/c++/8/ostream:38,
                 from /usr/include/c++/8/iostream:39,
                 from 1_3lab.cpp:2:
/usr/include/c++/8/bits/stringfwd.h:74:33: note:   ‘std::__cxx11::’string
   typedef basic_string<char>    string;
                                 ^~~~~~
1_3lab.cpp:11:2: error: ‘’cin was not declared in this scope
  cin >> tmp;
  ^~~
1_3lab.cpp:11:2: note: suggested alternative:
In file included from 1_3lab.cpp:2:
/usr/include/c++/8/iostream:60:18: note:   ‘std::’cin
   extern istream cin;  /// Linked to standard input
                  ^~~
1_3lab.cpp:11:9: error: ‘’tmp was not declared in this scope
  cin >> tmp;
         ^~~
1_3lab.cpp:11:9: note: suggested alternative: ‘’tm
  cin >> tmp;
         ^~~
         tm
1_3lab.cpp:12:2: error: ‘’cout was not declared in this scope
  cout << tmp << endl;
  ^~~~
1_3lab.cpp:12:2: note: suggested alternative:
In file included from 1_3lab.cpp:2:
/usr/include/c++/8/iostream:61:18: note:   ‘std::’cout
   extern ostream cout;  /// Linked to standard output
                  ^~~~
1_3lab.cpp:12:17: error: ‘’endl was not declared in this scope
  cout << tmp << endl;
                 ^~~~
1_3lab.cpp:12:17: note: suggested alternative:
In file included from /usr/include/c++/8/iostream:39,
                 from 1_3lab.cpp:2:
/usr/include/c++/8/ostream:590:5: note:   ‘std::’endl
     endl(basic_ostream<_CharT, _Traits>& __os)
     ^~~~
```

Листинг 4 – ошибка компиляции

Задание 2

Задание 2. *Написать программу, определяющую различные варианты функции plus: обычное сложение и сложение по модулю 2. Использовать механизм пространств имен.*

Программа 5 компилируется и исполняется без ошибок, как оджидается. Создаются два namespace'а (*boolean* для сложения по модулю 2 и *common* для обычного). Программа принимает любое кол-во аргументов. Если был получен аргумент "c" выполняется *common::plus*, если "b", то *boolean::plus*. Если аргументов не было переданно, то выполняется *common::plus*. Для удобства вызова был реализован макрос *PRINT*. Макрос задает функцию *print* принимающую из стандартного вввода два аргумента и передает их в функцию *plus*, результат которой выводится в стандартный вывод в удобном формате. Макрос используется внутри каждого пространства имен, что приводит к тому, что внутри вызывается функция *plus* соответствующего пространства. В функции *main* вызываются *boolean::print* и *common::print*.

```
#include <iostream>
#include <cstring>

#define PRINT(type, symbol) \
  void print() \
  { \
    type a, b; \
    std::cin >> a >> b; \
    std::cout << a << " " #symbol " " << b << " = " << plus(a, b) << std::endl; \
  }

namespace boolean {
  inline bool plus(bool a, bool b)
  {
    return (!a || !b) && (a || b);
  }

  PRINT(bool, (+))
};

namespace common {
  inline int plus(int a, int b)
  {
    return a + b;
  }

  PRINT(int, +)
}

#undef PRINT



int main(int argc, char** argv)
{
  //c — common, b — mod 2
  const std::string c("c"), b("b");
  if (argc > 1) {
    for (int i = 1; i < argc; ++i) {
      std::string tmp(argv[i]);
      if (tmp == c)
        common::print();
      else if(tmp == b)
        boolean::print();
    }
  } else
    common::print();


}
```

Листинг 5 – Функции plus

Задание 3

Задание 3. *Написать процедуру, изменяющую значение своего аргумента, используя:*

- *передачу параметра по значению;*
- *передачу параметра по ссылке;*
- *указатели;*

Вывести на экран значение аргумента до и после вызова процедур. Поясните результат. Чем отличается синтаксис вызова процедур?

Программа 6 компилируется и исполняется без ошибок, как оджидается. Для удобства был реализован макрос *FOO* , который позволяет удобно задавать функцию по изменению аргумента, обеспечивая при этом индикацию вызова функции в стандартный вывод. Пример работы программы 7 . Видно, что только функция *foo0(int a)* не изменила аргумент, что ожидаемо, потому что данная функция принимает аргумент по значению. Другие же две фунции принимают аргумент по ссылке и указателю, что позволяет менять значение аргумента в теле функции.

```
#include <iostream>

#define FOO(name, type, operation) \
  inline void name(type a) \
  { \
    std::cout << #name << "(" #type " a)" << std::endl; \
    operation; \
  }

FOO(foo0, int, a++)
FOO(foo1, int&, a++)
FOO(foo2, int*, (*a)++)


#undef FOO

int main()
{
  int a;
  std::cin >> a;

#define PRINT(step) \
  std::cout << "step" #step " a = " << a << std::endl << std::endl;

  PRINT(0)

  foo0(a);
  PRINT(1)

  foo1(a);
  PRINT(2)

  foo2(&a);
  PRINT(3)

#undef PRINT

}
```

Листинг 6 – Процедуры изменяющие аргумент

```
0
step0 a = 0

foo0(int a)
step1 a = 0

foo1(int& a)
step2 a = 1

foo2(int* a)
step3 a = 2
```

Листинг 7 – Вызов программы

Задание 4

Задание 4. *Проведите компиляцию кода листинга 8 и проанализируйте выполнение программы инструментом Dr. Memory.*

- *объясните получаемые ошибки;*
- *напишите исправленную версию программы.*

Выполнение программы 8 было проанализированно с помощью программы Valgrid. Было обнаружено 3 утчеки в общей сумме на 12 байт. При вызгляде на код прогаммы 8 становится очевидно, что вся память выделенная *new* не была освобождена. Поэтому была написанна другая программа 10 , у которой проблем с утечкой памяти нет, как ожидается.

```
#include <stdlib.h>
#include <iostream>

using namespace std;

void fv() {
  int i = int(123);
  cout << "i = " << i << endl;
}

void fp1()
{
  int* i = new int(123);
  cout << "*i = " << *i << endl;
}

void fp2()
{
  int* i = new int(123);
  cout << "*i = " << *i << endl;
  i = new int(456);
  cout << "*i = " << *i << endl;
}

int* fr()
{
  int i = int(123);
  int* ip = &i;
  cout << "i = " << i << endl;
  return ip;
}

int main()
{
  fv();
  fp1();
  fp2();
  int *ifr = fr();
  cout << "*ifr = " << *ifr << endl;
  return 0;
}
```

Листинг 8 – Вызов программы

```
==14048== Memcheck, a memory error detector
==14048== Copyright (C) 2002−2017, and GNU GPL'd, by Julian Seward et al.
==14048== Using Valgrind−3.14.0 and LibVEX; rerun with −h for copyright info
==14048== Command: ./a.out
==14048==
==14048== Conditional jump or move depends on uninitialised value(s)
==14048==    at 0x49713CB: std::ostreambuf_iterator<char, std::char_traits<char> > std::num
==14048==    by 0x497CFB4: std::ostream& std::ostream::_M_insert<long>(long) (in /usr/lib/x8
==14048==    by 0x10936C: main (in /home/shurik/Work/studies/10sem/oop/1lab/a.out)
==14048==
==14048== Use of uninitialised value of size 8
==14048==    at 0x4970EDE: ??? (in /usr/lib/x86_64−linux−gnu/libstdc++.so.6.0.25)
==14048==    by 0x49713F4: std::ostreambuf_iterator<char, std::char_traits<char> > std::num
==14048==    by 0x497CFB4: std::ostream& std::ostream::_M_insert<long>(long) (in /usr/lib/x8
==14048==    by 0x10936C: main (in /home/shurik/Work/studies/10sem/oop/1lab/a.out)
==14048==
==14048== Conditional jump or move depends on uninitialised value(s)
==14048==    at 0x4970EEB: ??? (in /usr/lib/x86_64−linux−gnu/libstdc++.so.6.0.25)
==14048==    by 0x49713F4: std::ostreambuf_iterator<char, std::char_traits<char> > std::num
==14048==    by 0x497CFB4: std::ostream& std::ostream::_M_insert<long>(long) (in /usr/lib/x8
==14048==    by 0x10936C: main (in /home/shurik/Work/studies/10sem/oop/1lab/a.out)
==14048==
==14048== Conditional jump or move depends on uninitialised value(s)
==14048==    at 0x4971427: std::ostreambuf_iterator<char, std::char_traits<char> > std::num
==14048==    by 0x497CFB4: std::ostream& std::ostream::_M_insert<long>(long) (in /usr/lib/x8
==14048==    by 0x10936C: main (in /home/shurik/Work/studies/10sem/oop/1lab/a.out)
==14048==
==14048==
==14048== HEAP SUMMARY:
==14048==     in use at exit: 12 bytes in 3 blocks
==14048==   total heap usage: 5 allocs, 2 frees, 73,740 bytes allocated
==14048==
==14048== LEAK SUMMARY:
==14048==    definitely lost: 12 bytes in 3 blocks
==14048==    indirectly lost: 0 bytes in 0 blocks
==14048==      possibly lost: 0 bytes in 0 blocks
==14048==    still reachable: 0 bytes in 0 blocks
==14048==         suppressed: 0 bytes in 0 blocks
==14048== Rerun with −−leak−check=full to see details of leaked memory
==14048==
==14048== For counts of detected and suppressed errors, rerun with: −v
==14048== Use −−track−origins=yes to see where uninitialised values come from
==14048== ERROR SUMMARY: 8 errors from 4 contexts (suppressed: 0 from 0)
```

Листинг 9 – Вызов программы

```
#include <stdlib.h>
#include <iostream>

using namespace std;

void fv() {
  int i = int(123);
  cout << "i = " << i << endl;
}

void fp1()
{
  int* i = new int(123);
  cout << "*i = " << *i << endl;
  delete i;
}

void fp2()
{
  int* i = new int(123);
  cout << "*i = " << *i << endl;
  delete i;
  i = new int(456);
  cout << "*i = " << *i << endl;
  delete i;
}

int* fr()
{
  int i = int(123);
  int* ip = &i;
  cout << "i = " << i << endl;
  return ip;
}

int main()
{
  fv();
  fp1();
  fp2();
  int *ifr = fr();
  cout << "*ifr = " << *ifr << endl;
  return 0;
}
```

Листинг 10 – Вызов программы