# Programming in C++: Assignment Week 4

Total Marks : 20

August 12, 2017

## Question 1

Which of the following operators can use friend functions for overloading? *Mark 1*

a. ==

b. [ ]

c. ->

d. ( )

**Answer:** a
**Explanation:** As per language syntax, check slides

## Question 2

What will be the O/P of the following program ? *Mark 1*

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

class GlobalClass {
    int m_value;
    static GlobalClass *s_instance;
    GlobalClass(int v = 0) {
        m_value = v;
    }
public:
    int get_value() {
        return m_value;
    }
    void set_value(int v) {
        m_value = v;
    }
    static GlobalClass *instance() {
       if (!s_instance)
       s_instance = new GlobalClass;
       return s_instance;
    }
};
```

```
GlobalClass *GlobalClass::s_instance = 0;

void Func1(void) {
    GlobalClass::instance()->set_value(1);
    cout  << GlobalClass::instance()->get_value() << '\n';
}

void Func2(void){
    GlobalClass::instance()->set_value(2);
    cout <<  GlobalClass::instance()->get_value() << '\n';
}

int main() {
    cout << GlobalClass::instance()->get_value() << '\n';
    Func1();
    Func2();
}
```

a. 0 1 1

b. 0 0 0

c. 0 1 2

d. 1 2 3

**Answer:** c

**Explanation:** Defining singleton objects keeps only one copy of it, to be accessed by all functions

## Question 3

What is the output of the following code? *Mark 1*

```
#include <iostream>
using namespace std;
struct emp {
    int a;
    emp ( int b): a(b){cout << " Constructor " ;}
   ~emp(){ cout << " Destructor " ;}
    void disp(){ cout << " In Display " ; }
};

int main(){
    emp *e =  new emp(20);
    cout << e->a ;
    e->disp();
}
```

Output of the Code is

a. `a` and `disp` cannot be accessed in main as it is private

b. `a` cannot be accessed in main as it is private

c. Constructor 20 In Display

d. Constructor 20 In Display Destructor
**Answer:** c
**Explanation:** As per execution semantics of structures, where there is no access specifier

## Question 4

What is the output of the following code? *Mark 1*

```
#include <iostream>
using namespace std ;
namespace Ex { int x = 10; }
namespace Ex { int y = 10; }
int x = 5;
int main(){
    using namespace Ex ;
    x = y = 50;
    cout << x << " " << y;
}
```

a. 10 10

b. 50 50

c. 5 50

d. Compilation error: ambiguous reference to variable 'x'
**Answer:** d
**Explanation:** Ambiguity due to namespace defintion as well as global variable declaration

## Question 5

Fill in the blank. *Mark 1*

```
#include<iostream>
using namespace std;
class Test { static int x;
    public:
    void get() { x = 15; }
    void print() {
    x = x + 20;
    cout << "x =" << x << endl;
    }
};
____________; // Define static variable 'x'
int main() {
    Test o1, o2;
    o1.get(); o2.get();
    o1.print(); o2.print();
    return 0;
}
```

a) int Test t.x = 0;

b) Test t; t.x = 0;

c) int Test::x = 0;

d) Test t; t::x = 0;

**Answer**: c)
**Explanation:** Static variables are declared and initialised with class name, check slides

## Question 6

What will be the output of the following program? *Mark 1*

```
#include<iostream>
using namespace std;
class Test { int x;
    public:
    Test(int i) : x(i) {}
    friend void print(const Test& a);
};
void print(const Test& a) {
    cout << "x = " << a.x;
}
int main(){
    Test t(10);
    print(t);
    return 0;
}
```

a) x = 10

b) Compilation Error: print cannot access x as it is private

c) Compilation Error: illegal parameter passing in print

d) Compilation Error: Const parameter cannot be passed in friend function

**Answer**: a)
**Explanation:** x can be accessed as print is a friend function

## Question 7

What will be the output of the following program? *Mark 2*

```
#include <iostream>
using namespace std;
class sample {
    public:
    int x, y;
    sample() {};
    sample(int, int);
    sample operator + (sample);
};
sample::sample (int a, int b) {
    x = a;
    y = b;
```

```
}
sample sample::operator+ (sample param) {
    sample temp;
    temp.x = x + param.x;
    temp.y = y + param.y;
    return (temp);
}
int main () {
    sample a (4,1);
    sample b (3,2);
    sample c;
    c = a + b;
    cout << c.x << " " << c.y;
    return 0;
}
```

a) 5 5

b) 7 3

c) 3 7

d) 4 6

**Answer**: b)
**Explanation:** using operator overloading of + with class Sample objects

## Question 8

What will be the output of the following program? *Mark 2*

```
#include <iostream>
using namespace std;
class Test {
    int i;
    public:
    Test(int ii) : i(ii) {}
    const Test operator*(const Test& rv) const {
        cout << "Executes *" << endl;
        return Test(i * rv.i);
    }
    Test& operator+=(const Test& rv) {
       cout << "Executes +=" << endl;
       i += rv.i;
       return *this;
    }
};
int main() {
    int i = 1, j = 2, k = 3;
```

```
    k += i * j;
    Test ii(1), jj(2), kk(3);
    kk += ii * jj;
}
```

a) Executes *
   Executes +=

b) Executes *
   Executes +

c) Executes +=
   Executes *

d) Compilation Error: Ambiguous declaration

**Answer**: a)
**Explanation:** As per precedence of operators

## Programming Assignment

## Question 1

Fill in the blank below by writing the appropriate operator function, parameters and return type so that the given test cases will be satisfied. *Marks 2*

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


class Complex { double re, im; public:
     explicit Complex(double r = 0, double i = 0) : re(r), im(i) { }
     void disp() { cout << re << "+j" << im << endl; }
     friend Complex operator+ (const Complex &a, const Complex &b) {
          return Complex(a.re + b.re, a.im + b.im);
     }
     friend Complex operator+ (const Complex &a, double d) {
         Complex b(d); return a + b;
     }
     -------------------------- {
         Complex a(d); return a + b;
    }
};


int main(){
    double x, y, z, w;
    cin >> x >> y >> z >> w;;

    Complex d1(x, z), d2(y, w), d3;

    d3 = d1 + d2; d3.disp();
    d3 = d1 + 6.2; d3.disp();
```

```
    d3 = 4.2 + d2; d3.disp();

    return 0;
}
```

**Answer:** friend Complex operator+ (double d, const Complex &b)
**Explanation:** Operator function to take a double number and a complex data type in order

a. Input:

```
3.4
5.6
 6
 7

Output:
 9 + j 13
 9.6 + j 6
 9.8 + j 7
```

b. Input:

```
5
7
4
5

Output:
     12 + j 9
     11.2 + j 4
     11.2 + j 5
```

c. Input:

```
0
1
1
1

Output:
1 +j 2
6.2 +j 1
5.2 +j 1
```

## Question 2

Here S and R Represent two geometric class, Square and Rectangle respectively. Our objective is to convert /Interpret the Square object as Rectangle and calculating the area of rectangle. *Marks 2*

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

class S;

class R {
    int width, height;
    public:
    int area ()   // Area of rectangle
   {return (width * height);}
    void convert (S a);
};

class S {
   ____________________;  // Fill the blank
   private:
   int side;
   public:
   S (int a) : side(a) {}
};

void ____________________ (S a) {
   width = a.side;
   height = a.side;    // Interpreting Square as an rectangle
}

int main () {

   int x = 4;

   cin >> x;
   R rect;
   S sqr (x);
   rect.convert(sqr);
   cout << rect.area();
   return 0;
}
```

**Answer:** friend class R// R::convert
**Explanation:** If a class needs to access the private members(width and height) of a different class, it should be a declared as a friend class.

a. Input: 4

```
Output: 16
```

b. Input: -6

```
Output: 36
```

c. Input: -2.5

```
Output: 4
```

## Question 3

This Program is all about the implementation of Pre/Post Incrementer. Fill the blank By keeping this in mind so that the given test cases will satisfy. *Marks 2*

```
#include <iostream>
using namespace std;
class MyClass { int data;
    public:
    ____________________{ }  // Define Constructor
    MyClass& operator++() {
        ++data;
        return ___________;
    }
   ____________________________ {
   MyClass t(data);
       ++data;
       return _______________;
   }
   void disp() { cout << " " << data ; }
};
int main() {

    int x;

    cin >> x;
    MyClass obj1(x);
    obj1.disp();
    MyClass obj2 = obj1++;
    obj2.disp();
    obj2 = ++obj1;
    obj2.disp();
    return 0;
}
```

**Answer:** MyClass(int d): data(d) // *this // MyClass operator++(int) // t //
**Explanation:** As per operational semantics of the post and pre increment operators, check slides.

a. Input: 4

```
Output: 4 4 6
```

b. Input: -9

```
Output: -9 -9 -7
```

c. Input: 0

```
Output: 0 0 2
```

## Question 4

Here display() is a function of `YourClass` which should display the data member of Myclass. Add the required code in editable section to satisfy our objective. *Marks 2*

```
#include<iostream>
using namespace std;
class MyClass { int x_;
public:
    MyClass(int i) : x_(i) {}

};

class YourClass { int y;
public:
    void display(const MyClass &a) {
        cout << " " << a.x_;
    }

};
int main(){

    int x;

    cin >> x;
    MyClass obj(x);
    YourClass y;
    y.display(obj);
    return 0;
}
```

**Answer:** Add the code "friend class YourClass;" after the constructor of Myclass.
**Explanation:** To access the private member of a class, a non member function(in this case display) should be a declared as a friend function. Check the slides

a. Input: 4

```
Output: 4
```

b. Input: 8.7

```
Output: 8
```

c. Input: 0

```
Output: 0
```

## Question 5

Fill the blank by keeping in mind that, the program tests the conceptual knowledge about static*Marks 2*

```
#include<iostream>
using namespace std;
class MyClass { static int x;
    public:

    void get() { x++; }
    _______ ________ print(int y) { //Fill the blank with proper key words
        x = x - y;
        cout << " " << x ;
   }
};
_______________________;  // Define static data member
int main() {

   int x;

   cin >> x;
   MyClass:: print(x);
   MyClass o1;
   o1.get();
   o1.print(x);
   return 0;
}
```

**Answer:** static void // int MyClass::x = 1
**Explanation:** Static variables can be initialised outside the scope of the main without constructing objects. It remains live outside main.

a. Input: 5

```
Output: -4  -8
```

b. Input: 0

```
Output: 1 2
```

c. Input: -7

```
Output: 8 16
```