Assignment On

Submitted To: Ahsanul Haque

Submitted By: Md. Zobayer Hasan Nayem

Id: 19202103274

Section: 7

Intake: 44

Course: CSE-121

Bangladesh University Of Business and Technology

Md. Zobayer Hasan Nayem

ID : 19202103274

Section : 7

Intake : 44

Gmail : zobayer.hp3@gmail.com

Code Name: C++ program for static data member

```
# include <iostream>
using namespace std;
Class Item {

Static int countNum;
int number;
public:
 void getdata (int a)
 {
   number = a;
  count Num ++;

 }
 void getcount (void)
 {
 cout << "count :" << countNum << "\n";

 };
 int Item :: countNum;
 int main ()
 {
```

```
Item a, b, c
a.getcount ();
b.getcount ();
c.getcount ();

a.getdata (100);
b.getdata (100);
c.getdata (100);
cout << "After reading data" << "\n";

a.getcount ();
b.getcount ();
c.getcount ();
return 0;
}
```

code Name: Static member function

```
#include <iostream>
using namespace std;
class Test{
int code;
static int count;
public :
void setcode (void)
{
    code=++count;
}
void showcode (void)
{
   cout << "object number: " <<code << end1;
}
 static void showCount (void) // static member fu
{
 cout << "count:" << count <<end1 ;
 }
};
```

```
int Test : : count;
int main ()
{
Test t1, t2;
t1. setcode ();
t2. setcode ();
Test : : showcount (); // accessing static function
Test t3;
t3.setCode ();
Test :: showCount ();
t1. showCode ();
t2. showCode ();
t3. showCode ();
return 0;
}
```

code Name : Objects as function arguments

```
# include <iostream>
using namespace std;
class Time {
  int hours;
  int minutes;
  public:
  void getTime (int h, int m)
  {
   hours = h;
   minutes = m;

  }
  void putTime ()
  {
   cout << hours << "hours and";
   cout << minutes << "minutes" << "\n";
  }
    void sum (Time, Time);
 };
void Time :: sum ( Time t1, Timt2)
 {
```

```
minutes = t1 . minutes + t2 . minutes;
hours = minu-tes /60;
minutes = minutes %o 60;
hours = hours + t1 . hours + t2 . hours;
}

int main ()
{
Time T1 , T2 , T3;
T1. getTime (2,45);
t2. getTime (3,30);

T3. sum(T1 ,T2);
cout << "T1= ";
T1.putTime ();
cout <<"T2 =";
T2. putTime ();
cout <<"T3 =";
T3. put Time ();
return 0;
}
```

Code Name : Friendly Function

```
# include <iostream>
using namespace std;
class Sample {
int a;
int b;
public:
void setValue(){a=25; b=40;}
friend float mean (Sample s);
};
float mean (Sample s)
{
 return float (s.a + s.b) /2.0;
}
int main ( )
{
 Sample X;
X. setValue();
cout << "Mean value = " << mean (x) << "\n";
return 0;
}
```

code Name : Returning objects

```
# include <iostream>
 using namespace std;
class complex {  //
float x;        //
float y;        //
public :
void input (float real, float imag)
{
 x = real;
 y = imag;
}
friend complex sum (Complex c1, Complex c2);
void show (complex);
};
Complex sum (complex c1, Complex c2)
{
Complex c3;
C3.x = C1.x + C2.x;
C3.y = C1.y + C2.y;
return . C3;
}
void Complex : : show (complex c)
{
cout << c.x << " + " << c.y << end1;
```

```
int main ()
{
Complex A, B, C;

A. input (3.1, 5.65);
B. input (2.75, 1.2);
C = sum (A, B); // C = A + B.
cout << "A = "; A. show (A);
cout << "B = "; B. show (B);
cout << "C = "; C. show (C);
return 0;

}
```

code Name: A function friendly to two classes

```
#include <iostream>
using namespace std;
class ABC;
class XYZ {
  int x;
  public:
  void setvalue (int i) {x = i; }
  friend void max (XYZ,ABC);

};
void max( XYZ m, ABC n)
{
   if (m.x >= n.x)
     cout << " Max :" << m.x;
   else
     cout <<"Max :" << n.x;

 }
 int main ()
 {
  ABC abc;
  abc.setValue (10);
  XYZ xyz;
  xyz. set Value(20);
  max (xyz, abc);
   return 0;
}
```

code name : Parameterized constructors

```
# include <iostream>
using namespace std;
class integer
{
 int m,n;
public:
integer (int x, int y); //constructor declared

 void display (void)
 {
 cout <<" m = " << m << "\n ";
 cout << "n  = " << n << "\n";

 }

};
integer :: integer (int x, int y) // constructor defined
{
m = x; n = y;
}
int main ( )
{
```

```
integer int1 = integer(50,100);
integer int2 (25,75);
cout << "\nOBJECT1" << "\n" ;
int1. display ();
cout << "\nOBJECT2" << "\n";
int 2. display ();
return 0;
}
```

code name: Multiple constructor in a class

```
# include <iostream>
using namespace std;
Class Integer
{
int m,n;
public:
Integer () {m=0; n=0;}
Integer (int x, int y)
{
 m = x;
 n=y;
}
Integer (Integer &i )
{
 m = i.m;
 n= i.n;
}
void display (void)
{
  cout <<"m ="<< m <<",";
  cout <<"n =" << n <<"\n";
}
};
```

code name: Multiple constructor in a class

```
# include <iostream>
using namespace std;
class Complex
{
float x, y;
public :
complex () { }
complex (float a) {x=y=a;}
Complex (float real, float imag)
{x = real; y= imag;}
friend Complex sum (complex ,Complex);
friend void show (complex);
};
Complex sum (complex c1, Complex c2)
{
  complex c3;
  c3.x = c1.x +c2.x;
  c3.y = c1.y+c2.y;
  return c3;
}
```

```
void show (complex c)
{
cout << c.x <<" +j " <<c.y << end1;
}
int main ( )
{
complex A(2.7, 3.5);
Complex B(1.6);
Complex c;
c = sum(A,B);
cout <<"A = "; show(A);
cout <<"B = "; show(B);
cout <<"c = "; show(c);
// Another way to give initial value
Complex P, Q, R;
P = Complex (2.5, 3.9);
Q = Complex (1.6, 2.5);
R = sum (P,Q);
cout << "\n";
cout << "P = "; show (P);
Cout << "Q = "; show (Q);
cout << "R = "; show (R);
return 0;
}
```

code name: Nesting of member functions

```
# include <iostream>
using namespace std;
int m, n;
public:
void input (void);
void display (void);
void largest (void);
};
void set :: largest ()
{
   if (m >= n)
        cout << "Largest value = "<< m;

   else
        cout << "Largest value = " << n;

}

void set :: input (void)
{
    cout << "Input values of m and n" << end1;
    cin >> m >> n;
}
```

```
void Set :: display ()
{
   largest ();
}

int main ()
{
   Set A;
  A. input ();
  A. display ();
  return 0;
}
```

code Name : Nesting of member functions

```
#include <iostream>
using namespace std;
class Set {

  int m, n;
 public :
  void input (void);
  void display (void);
};
 void Set : : input (void)
 {
   cout << " Input values of m and n"<< end1;
   cin >> m >> n;

 }
void Set : : display (void)
{
   . cout << " Input values of
```

code name: Nesting of member functions

```
#include <iostream>
using namespace std;
class Set {
int m, n;
public:
void input (void);
void display (void);
};
void Set :: input (void)
{
  cout << Input Values of m and n" << end1;
  cin >> m >> n;

}
 void Set :: display( )
 {
 if (m>=n)
  cout << " Lagest value =" << m;
else
 cout << " Largest value =" << n;
}
```

```
int main ( )
{
 Set A:
 A.input ();
 A. display ();
 return 0;
}
```

code name : C++ program with class

```
#include <iostream.h>
class item
{
    int number : //variables declaration
    float cost   : //private by default

  public :
    void getdata (int a.float b);
    void putdata (void)
   {

        cout <<"Number :" << number <<"\n";
        cout <<" cost :" <<cost <<"\n";
   }
};
  void item :: getdata(int a. float b)
  {
      number = a;
      cost = b;
  }
void main ()
{
```

```
item x;                       // create object x
cout <<" \nobject x " << "\n ";
x.getdata (100. 299.95 );
x. put data ();

Item y;                    //create another object
y.getdata (200, 175.50);
y.putdata ();

}
```