

NAME  
UID  
BRANCH  
SUB  
SEMESTER  
D.O.P-

### EXPERIMENT NUMBER –Practical 3.1

**TOPIC OF EXPERIMENT** – Learn the basic of c++

**AIM OF THE EXPERIMENT** - Write a program to find the largest&smallest of three numbers. (Use inline function MAX and MIN)

### FLOWCHART/ALGORITHM

- ***Start.***
- ***Include the header file.***
- ***Add namespace to your program.***
- ***Define the two functions and pass the arguments to both the functions.***
- ***After each function apply the conditional statements which we will use to find the largest and the smallest number.***

- *Start the main function.*
- *Call both the functions.*
- *Call them by call by value method.*
- *Print the output.*
- *Stop.*

### PROGRAM CODE

```
#include <iostream>
using namespace std;
inline int Max(int x, int y, int z);
inline int Min(int x, int y, int z);
int main()
{
    int x, y ,z;
    cout<<"enter number:";
    cin>>x>>y>>z;
    int Maximum=Max(x,y,z);
    cout<< "Largest of " << x <<" , " << y << " , " << z << " is " <<Maximum;
    int Minimum=Min(x,y,z);
    cout<< "\n Smallest of " << x <<" , " << y << " , " << z << " is " << Minimum;
    return 0;
}
inline int Max(int x , int y , int z)
{if(x > y && x > z)
    return x;
    else if(y > z)
    return y;
    else
    return z;
}
inline int Min(int x , int y , int z) {
if(x < y && x < z)
    return x;
    else if(y < z)
    return y;
```

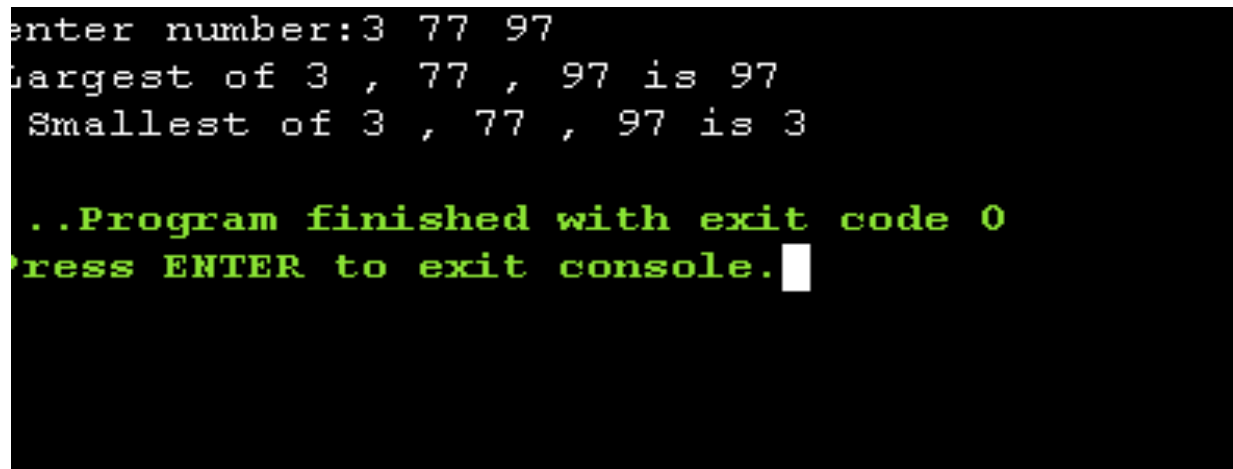
else

return z;

}



```
main.cpp
1 #include <iostream>
2 using namespace std;
3 inline int Max(int x, int y, int z);
4 inline int Min(int x, int y, int z);
5 int main()
6 {
7     int x, y, z;
8     cout<<"enter number:";
9     cin>>x>>y>>z;
10    int Maximum=Max(x,y,z);
11    cout << "Largest of " << x << " , " << y << " , " << z << " is " << Maximum;
12    int Minimum=Min(x,y,z);
13    cout << "\n Smallest of " << x << " , " << y << " , " << z << " is " << Minimum;
14    return 0;
15 }
16 inline int Max(int x , int y , int z)
17 {if(x > y && x > z)
18     return x;
19     else if(y > z)
20     return y;
21     else
22     return z;
23 }
24 inline int Min(int x , int y , int z) {
25     if(x < y && x < z)
26     return x;
27     else if(y < z)
28     return y;
29     else
30     return z;
31 }
```



```
enter number:3 77 97
Largest of 3 , 77 , 97 is 97
Smallest of 3 , 77 , 97 is 3

..Program finished with exit code 0
Press ENTER to exit console.
```

**ERRORS ENCOUNTERED DURING PROGRAM'S EXECUTION (Kindly jot down the compile time errors encountered)**

**No error**

### **Practical 3.2**

**A dining hall can accommodate only 50 guests. Create a class to store seat number (Generated Automatically) and name of the guests who are seated on first come first seated basis. Define functions to display name of all guests along with seat number. Write a program to show the working of this class using the concept of static data member and static function**

#### **FLOWCHART/ALGORITHM**

- *Start.***
- *Include the header file.***

- **Add namespace to your program.**
- **Define the maximum size of the hall by using the macro.**
- **Create the two classes guest and the hall.**
- **Declare the required variables. Inside the class. Declare the variables using static keyword also.**
- **After that apply the logic so that our program can specify the order of first come and first seat basis.**
- **Ask the user to input the names of all the guests.**
- **Using the scope resolution operator call the static variables.**
- **Now start the main function and call your class, apply the logic so that we can print the required output.**
- **Print the output.**
- **Stop.**

### **PROGRAM CODE**

```
#include <iostream>
#define MAX_SIZE 60
using namespace std;

class Guest {
public:
char name[ 60 ];
int seatno;
};
class Hall {public:
static int front, rear;
static Guest allGuest[MAX_SIZE];

static int alloteSeat()
{
if (rear == (MAX_SIZE - 1 ))
{
cout<< "Hall is FULL" ;
return 0 ;
```

```

}
rear++;
cout<< "Enter Guest Name:\n " ;
cin>>allGuest[rear].name;
allGuest[rear].seatno = rear + 1 ;
return 1 ;
}
static void listGuest() {
while (++front <= rear) {
cout<< "\nGuest " <<allGuest[front].name << " is seated on seat S"
<<allGuest[front].seatno<< "." ;
}
rear = front = - 1 ;
}
};
int Hall::front = - 1 ;
int Hall::rear = - 1 ;
Guest Hall::allGuest[MAX_SIZE] = {};
int main()
{
Hall::alloteSeat();
Hall::alloteSeat();
Hall::alloteSeat();
Hall::alloteSeat();
Hall::listGuest();return 0 ;
}

```

```

1 #include <iostream>
2 #define MAX_SIZE 60
3 using namespace std;
4
5 class Guest {
6 public:
7 char name[ 60 ];
8 int seatno;
9 };
10 class Hall {public:
11 static int front, rear;
12 static Guest allGuest[MAX_SIZE];
13
14 static int alloteSeat()
15 {
16 if (rear == (MAX_SIZE - 1 ))
17 {
18 cout<< "Hall is FULL" ;
19 return 0 ;
20 }
21 rear++;
22 cout<< "Enter Guest Name:\n " ;
23 cin>>allGuest[rear].name;
24 allGuest[rear].seatno = rear + 1 ;
25 return 1 ;
26 }
27 static void listGuest() {
28 while (++front <= rear) {
29 cout<< "\nGuest " <<allGuest[front].name << " is seated on seat 5"
30 <<allGuest[front].seatno<< "." ;
31 }
32 }

```

```

32 rear = front = - 1 ;
33 }
34 };
35 int Hall::front = - 1 ;
36 int Hall::rear = - 1 ;
37 Guest Hall::allGuest[MAX_SIZE] = {};
38 int main()
39 {
40 Hall::alloteSeat();
41 Hall::alloteSeat();
42 Hall::alloteSeat();
43 Hall::alloteSeat();
44 Hall::listGuest();return 0 ;
45 }

```

**ERRORS ENCOUNTERED DURING PROGRAM'S EXECUTION (Kindly jot down the compile time errors encountered)**

No error

## OUTPUT

```
Enter Guest Name:
a
Enter Guest Name:
b
Enter Guest Name:
c
Enter Guest Name:
d

Guest a is seated on seat S1.
Guest b is seated on seat S2.
Guest c is seated on seat S3.
Guest d is seated on seat S4.

...Program finished with exit code 0
Press ENTER to exit console. █
```

### Practical 3.3

WAP to swap private data members of classes named as class\_1, class\_2 using friend function.

#### FLOWCHART/ALGORITHM

- **Start.**
- **Include the header file.**
- **Add namespace to your program.**
- **create two classes 1 and 2.**



- *In both the classes create two variables num1 and num2 which will be of protected type.*
- *Initialize the variables with some value inside the member function.*
- *Between class one and two we will declare a function which will display the original values given to the num1 and num2.*
- *Then apply the friend function which will swap the values of num1 and num2.*
- *Now add the class2 to your code.*
- *Again apply the same friend function after the class2.*
- *Declare another function in which we will apply the logic to swap the numbers.*
- *Start main function and create object of class 1 and 2 respectively.*
- *Print the output.*
- *Stop.*

#### **PROGRAM CODE**

```
#include <iostream>
using namespace std;
class class_2;
class class_1
{
protected:
int num1;
public:
class_1()
{
```

```
num1= 10 ;
}
void show()
{
cout<< "\n Value of Class 1 : " <<num1;
}
friend void swap( class _1 *num1, class _2 *num2);
};
class class_2
{
protected:
int num2;
public:
class_2()
{
num2= 20 ;
}
void show()
{
cout<< "\n Value of Class 2 : " <<num2;
}
friend void swap( class _1 *num1, class _2 *num2);
};
void swap( class _1 *no1, class _2 *no2)
{
int no3;
no3=no1->num1;
no1->num1=no2->num2;
```

```
no2->num2=no3;
}
int main()
{
class_1 a;
class_2 b;
cout<< "\nValuesbefor Swap\n" ;
a.show();
b.show();
swap(&a, &b);
cout<< "Values after Swap\n" ;
a.show();
b.show();
return 0 ;
}
```

```

main.cpp
1 #include <iostream>
2 using namespace std;
3 class class_2;
4 class class_1
5 {
6 protected:
7 int num1;
8 public:
9 class_1()
10 {
11 num1= 10 ;
12 }
13 void show()
14 {
15 cout<< "\n Value of Class 1 : " <<num1;
16 }
17 friend void swap( class_1 *num1, class_2 *num2);
18 };
19 class class_2
20 {
21 protected:
22 int num2;
23 public:
24 class_2()
25 {
26 num2= 20 ;
27 }
28 void show()
29 {
30 cout<< "\n Value of Class 2 : " <<num2;
31 }
32 friend void swap( class_1 *num1, class_2 *num2);

```

```

33 };
34 void swap( class_1 *no1, class_2 *no2)
35 {
36 int no3;
37 no3=no1->num1;
38 no1->num1=no2->num2;
39 no2->num2=no3;
40 }
41 int main()
42 {
43 class_1 a;
44 class_2 b;
45 cout<< "\nValues befor Swap\n" ;
46 a.show();
47 b.show();
48 swap(&a, &b);
49 cout<< "Values after Swap\n" ;
50 a.show();
51 b.show();
52 return 0 ;
53 }

```

**ERRORS ENCOUNTERED DURING PROGRAM'S EXECUTION (Kindly jot down the compile time errors encountered)**

No error

**OUTPUT**

```
Values befor Swap  
  
Value of Class 1 : 10  
Value of Class 2 : 20  
  
Values after Swap  
  
Value of Class 1 : 20  
Value of Class 2 : 10
```

### Practical 3.4

WAP to create a class complex to represent complex numbers. The complex class should use a function to add two complex number which are passed as arguments. The function should return an object of type complex representing the sum of two complex numbers.

#### FLOWCHART/ALGORITHM

- **Start.**
- **Include the header file.**
- **Add namespace to your program.**
- **Create a class named complex. Declare two variables privately.**
- **Declare a function publicly and pass two arguments to the function and equalize them with the two variables we defined as privately.**
- **Add a member variable and pass an argument c to it.**
- **Add the formula to add the real part of the of both the complex numbers.**
- **Now declare another variable to show the different conditions of the addition of numbers. (like if  $i=0$  then what will happen)**
- **Start the main function and create the objects and give the values to the objects**
- **After that call the function set and disp one by one.**
- **Print the output.**

· **Stop.**

## PROGRAM CODE

```
#include <iostream>
using namespace std;
class complex
{
private:
float r;
float i;
public:
void set( float real, float img)
{
r = real;
i = img;
}
complex sum(complex c)
{
complex t;
t.r = r + c.r;
t.i = i + c.i;
return t;
}
void disp()
{
if (i == - 1 ) {
cout<< r << " + -i" <<endl;
}
else if (i == 1 ) {
cout<< r << " + i" <<endl;
}
else if (i == 0 ) {
cout<< r <<endl;
} else {
cout<< r << " + " <<i<< "i" <<endl;
}
}
};
int main()
```



```

{
complex c1, c2, c3;
c1.set( 7.5 , 6.5 );
c2.set( 0.5 , 5.5 );
c3 = c1.sum(c2);
cout<< "\nComplex Number 1 = \n" ;
c1.disp();
cout<< "\nComplex Number 2 = \n" ;
c2.disp();
cout<< "\nComplex Number 3 = \n" ;
c3.disp();
return 0 ;
}

```

```

main.cpp
1  #include <iostream>
2  using namespace std;
3  class complex
4  {
5  private:
6  float r;
7  float i;
8  public:
9  void set( float real, float img)
10 {
11 r = real;
12 i = img;
13 }
14 complex sum(complex c)
15 {
16 complex t;
17 t.r = r + c.r;
18 t.i = i + c.i;
19 return t;
20 }
21 void disp()
22 {
23 if (i == - 1 ) {
24 cout<< r << " + -i" <<endl;
25 }
26 else if (i == 1 ) {
27 cout<< r << " + i" <<endl;
28 }
29 else if (i == 0 ) {
30 cout<< r <<endl;
31 } else {
32 cout<< r << " + " <<i<< "i" <<endl;
33 }
34 }
35 };
36 int main()
37 {
38 complex c1, c2, c3;
39 c1.set( 7.5 , 6.5 );
40 c2.set( 0.5 , 5.5 );
41 c3 = c1.sum(c2);
42 cout<< "\nComplex Number 1 = \n" ;
43 c1.disp();
44 cout<< "\nComplex Number 2 = \n" ;
45 c2.disp();
46 cout<< "\nComplex Number 3 = \n" ;
47 c3.disp();
48 return 0 ;
49 }
50

```

**ERRORS ENCOUNTERED DURING PROGRAM'S EXECUTION (Kindly jot down the compile time errors encountered)**

No error

### OUTPUT

```
complex Number 1 =  
.5 + 6.5i  
  
complex Number 2 =  
.5 + 5.5i  
  
complex Number 3 =  
+ 12i  
  
..Program finished with exit code 0  
press ENTER to exit console. 
```

**EVALUATION COLUMN (To be filled by concerned faculty only)**

Sr	Paramet	Maxim M	M Obtai
	Worksheet Completion including writing learning objective/ Outcome		
	Post Lab Quiz		
	Student engagement in Simulation/ Performance/ Pre Lab Questions		
	Total		