



**National University of Computer & Emerging Sciences, Karachi**  
**Computer Science Department**  
**Spring 2021, Lab Manual - 06**



<b>Course Code: CL-217</b>	<b>Course: Object Oriented Programming Lab</b>
<b>Instructor(s):</b>	<b>Nida Munawar, Abeer Gouhar, Romasha Khurshid, M. Fahim, Sohail Afzal, Qaiser Abbas, Ali Fatmi</b>

## Contents:

Static Keyword

Constant Keyword

## const Keyword in C++

Constant is something that doesn't change. In C language and C++, we use the keyword `const` to make program elements constant. `const` keyword can be used in many contexts in a C++ program. It can be used with:

1. Variables
2. Class Data members
3. Class Member functions

## Constant Variables in C++

```
int main
{
    const int i = 10;
    const int j = i + 10;    // works fine
    i++;    // this leads to Compile time error
}
```

## Constant data Member:

### Example Code:

```
#include <iostream>

using namespace std;

class Const_demo
{
    public:
    int total_Score_1;
    int total_Score_2;
    const int kristine_score;
    public:
        Const_demo():kristine_score(200)
        {
            //cout<<kristine_score;
        }
    public:
    void find_greater_score(int score1,int score2)
    {
        total_Score_1=score1;
        total_Score_2=score2;
        if(    score1<kristine_score&&score2<kristine_score)

        {
            cout<<"Kristine scores are the highest";

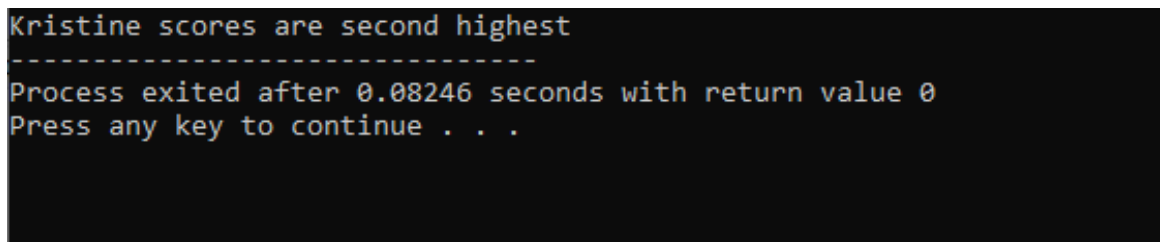
        }
        else if(score1>kristine_score&&score2>kristine_score)
```

```
{    cout<<"Kristine scores are third highest";
}

else
{
    cout<<"Kristine scores are second highest";
}
};

main()
{
    Const_demo cd;
    cd.find_greater_score(230,34);
}
```

## Output:



```
Kristine scores are second highest
-----
Process exited after 0.08246 seconds with return value 0
Press any key to continue . . .
```

## Constant Member Function:

### Example Code:

```
#include <iostream>

using namespace std;

class Const_demo
{
    public:
```

```
int total_Score_1;

int total_Score_2;
int kristine_score;
void input_Score()
{
    cin>>total_Score_1;
    cin>>total_Score_2;
    cin>>kristine_score;
}

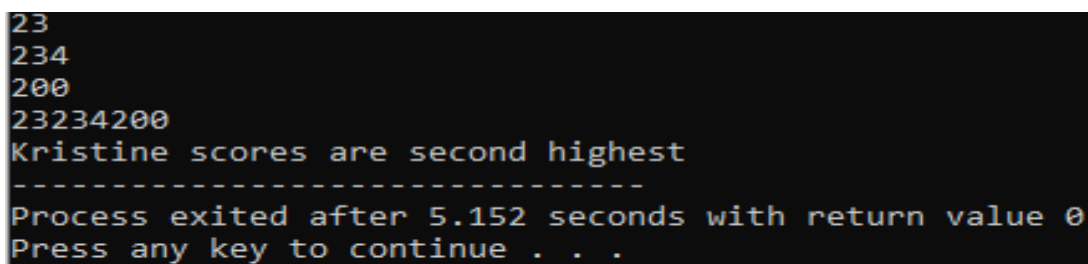
void change_score() const
{
    /*total_Score_1=2;
    total_Score_2=24; Produce Error
    kristine_score=200;*/

    cout<<total_Score_1<<total_Score_2<<kristine_score;
}

public:
    void find_greater_score()
    {
        if(    total_Score_1<kristine_score&&
total_Score_2<kristine_score)
            {
                cout<<"Kristine scores are the highest";
            }
    }
```

```
else if(total_Score_1>kristine_score&&total_Score_2>kristine_score)
{
    cout<<"Kristine scores are third highest";
}
else
    cout<<"Kristine scores are second highest";
}
};
main()
{
    Const_demo cd;
    cd.input_Score();
    cd.change_score();
    cd.find_greater_score();
}
```

## Output:



```
23
234
200
23234200
Kristine scores are second highest
-----
Process exited after 5.152 seconds with return value 0
Press any key to continue . . .
```

## Static Keyword in C++

Static is a keyword in C++ used to give special characteristics to an element. Static elements are allocated storage only once in a program lifetime in static storage area. And they have a scope till the program lifetime.

## Example Code:

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

void demo()
{
    // static variable
    static int count = 0;
    cout << count << " ";
    count++;
}
```

```
int main()
{
    for (int i=0; i<5; i++)
        demo();
    return 0;
}
```

## Static Data Member:

### Example Code:

```
#include<iostream>
using namespace std;
class static_demo{
public:
    static int a;
    void display()
    {
        cout<<"\n"<<a;
```

```
        }  
};  
  
int static_demo::a=10;  
main()  
{  
    static_demo::a=20;  
    cout<<"a="<<static_demo::a;  
    static_demo st;  
    st.display();  
}
```

## Output:

```
a=20  
20  
-----  
Process exited after 0.4244 seconds with return value 0  
Press any key to continue . . .
```

## Static Member Function:

### Example Code:

```
#include<iostream>

using namespace std;

class static_demo
{
    public:
        static int a;
        static void display()
        {
            cout<<"\n\t Hi meet me I am a static member function\n\t having
Static data memeber a="<<a;
        }
};

int static_demo::a=10;

main()
{
    static_demo st;
    cout<<"\n calling static member function using object name";
    st.display();
    cout<<"\n calling static member function using class name";
    static_demo::display();
}
```

### Output:

```
calling static member function using object name
    Hi meet me I am a static member function
    having Static data memeber a=10
calling static member function using class name
    Hi meet me I am a static member function
    having Static data memeber a=10
-----
```



## Practice Questions

1. Create a class Circle that contains a private data member radius. The class also contains a public data member static count which counts the number of objects created. Declare a parameterized constructor in the class which takes the radius as the parameter. The function getArea does not take any parameters and returns the area of the circle. In the main method create objects for the class Circle, display the total number of objects created and the area for each of those objects.
2. Create a class Demo that contains a functions showMessage. The function does not take any parameters and does not return a value. It only displays a message "Inside show message function". The class also contains a constant function display which does not take any parameters and does not return a value. It displays a message "Inside the display function". In the main function create objects of the class Demo and use both the functions.
3. Create a class called ADD that has two public static integer member variables named "a" and "b", and a public static member function named sum() that has no arguments but adds the two member variables together and returns their sum.
4. Write a class called CoffeeShop, which has two data members and seven member functions:

Constant public Name: a string (basically, of the shop)

Public Menu: an array of items.

addOrder: adds the name of the item to the end of the orders array if it exists on the menu. Otherwise, return "This item is currently unavailable!"

fulfillOrder: if the orders array is not empty, return "The {item} is ready!". If the orders array is empty, return "All orders have been fulfilled!"

listOrders: returns the list of orders taken, otherwise, an empty array.

dueAmount: returns the total amount due for the orders taken.

cheapestItem: returns the name of the cheapest item on the menu.

drinksOnly: returns only the item names of type drink from the menu.

foodOnly: returns only the item names of type food from the menu.