Assignment – 2

M.C.Q

Ans→ (1) d. Stack

Ans→(2) (c) Compiler Error in the line "Derived *dp = new Base;"

Ans→(3) a. Inaccessible.

Ans→ (4) (a) Programmer have to always call destructor at the end of the program.

Ans→ (5) True.

Short Answer type question-

Ans→ 1. Delete is an operator that is used to destroy array and non-array (pointer) objects which are created by new expression.

New operator is used for dynamic memory allocation which puts variables on heap memory.

Ex:→ (i) int *ptr = NULL;
ptr = new int ();

(ii) delete [] myarray;

Ans-12) A constructor is a special type of function with no return type. We define a method inside the class and constructor is also defined inside a class.

We required constructors to initialize the new object, that is, they set the startup property values for the object.

Types of constructors.

(1) Default Constructor :-)
It is the constructor which doesn't take any argument. It has no parameter.

ex:-)
```
class cube
  {
    public:
    int side;
    cube()
    {
      side = 10;
    }
  };
```
output = 10

(2) Parameterized Constructor :-)
Using this constructor you can provide different values to data members of different objects.

Ex:→
```
class Cube
{
    public:
    int side;
    Cube(int x)
    {
        side = x;
    }
};
int main()
{
    Cube c1(10);
    Cube c2(20);
    Cube c3(30);

    cout << c1.side;
    cout << c2.side;
    cout << c3.side;
```

Output : 10
20
30

(3) Copy constructor :→

It takes an object as argument, and is used to copy values of data members of one object into other object.

Ans-1 Difference b/w Procedual programming and object oriented programming :-

(1) Procedual oriented programing :-

(a) Program is divided into small parts called function.

(b) It follow top-down approach.

(c) There is no access specifier in it.

(d) Adding new data and function is not easy.

(2) Object oriented programming.

(a) Program is divided into small parts called objects.

(b) It follows bottom-up approach.

(c) It have access specifier like private, public, protected, etc.

(d) Adding new data and function is easy.

Long type Question :-

Ans→(1) Types of Polymorphism :-

(i) Compile time polymorphism
This type of polymorphism is achieved by function overloading or operator overloading :-

(a) Function overloading :-
When there are multiple functions with same name but different parameters then these functions are said to be overloaded.

(b) Operator overloading :-
C++ also provide option to overload operators.
For ex :- we can make the operator ('+') for string class to concatenate two string :-

(2) Runtime polymorphism :-
This type of polymorphism is achieved by Function overriding :- and it occur when a derived class has a definition for one of the member functions of the base class.

Code :-

```
#include <bits/stdc++.h>
using namespace std;
class base
{
  public:
    virtual void print()
{ cout << " print base class"
        << endl ; }

void show()
{ cout << " show base class" <<
        endl ; }
};
class derived : public base.
{
  public:
    void print()
{ cout << " print derived class.

}
return 0;
}
```

Ans-(2) #include <bits/stdc++.h>
using namespace std;

```
Void sort012(int a[], int arr_size)
{
    int lo = 0;
    int hi = arr_size - 1;
    int mid = 0;

    while (mid <= hi) {
        switch (a[mid]) {

        case 0:
            swap(a[lo++], a[mid++]);
            break;

        case 1:
            mid++;
            break;

        case 2:
            swap(a[mid], a[hi--]);
            break;
        }
    }
}
```

Function to print array arr[]

```
id print array (int arr[], int
    arr_size)
```

```
{
    for (int i=0; i< arr_size; i++)
        cout << arr[i] << " ";
}
int main()
{
    int arr[] = { 0, 1, 1, 0, 1, 2, 1, 2, 0
                , 0, 0, 1 3 };
int n = sizeof(arr) / sizeof(arr[0]);
sort 01 2 (arr, n);
cout << "array after segregation";
    print Array (arr, n);
    return 0;
}
```

Output :->

0 0 0 0 0 1 1 1 1 1 2 2