Assignment -2

Choose the Correct Option

1. Stack
2. Compiler error in line "Base *bp = new Derived;"
3. I naccessible
4. The number of times of destructor is called depends on Number of object creates.
5. True

Short answer type question.

1. Explain ~~about~~ new and delete keyword with code.

Ans ⇒ New keyword

The new operator is an operator which denotes a request for memory allocation on the heap. If Sufficient memory is available, a new operator initializes the memory and returns the address of the newly allocated and initialized memory to the pointer variable. When you create and object of class using new keyword. The constructor of the class is invoked to properly initialize this memory.

```
using namespace Std ;
 class car
   { String name ;
      int num ;
      Public:
       Car (String a, int n)
       { Cout << "Constructor called" << endl;
          this → name = a;
          this → num = n;
       }
       void enter()
        {    cin >> name;
             cin >> num;  }
void display()
 {
   Cout << "Name:" << name << endl;
   Cout << " Num; << num << endl;
 }
 };
 int main ()
 {  Car * P = new Car ("Hond", 20281);
     p -> display ();
 }
```

Delete keyword

Delete is an operator that is used to destroy array and non array object which are created by new expressions. Delete can be used by either using delete operator or Delete [] operator.

```
# include <bits/stdc++.h>
  using namespace Std;
  int main ()
  {  int * array = new int[n];
      delete [] array;
      return 0;
  }
```

2. What are Constructors? why they are required? Explain different types of Sonstuctors with Suitable example.

Ans 2) A constructor is a member function of a class which initializes object of a class. In C++, constructor is automatically called when object Create. It is Special member function of the class.

Types of constructors:-

a.) Default Constructors:

Default Constructors are the Constructor which doesn't take any argument. It has no parameters.

b.) Paramaetized Constructors:

It is possible to pass arguments constructors. Typically, these argument help initialize an object when it is created. To create parameterized constructors, Simply add parameters to it the way you would to any other function. When you define the constructors body use the parameters to initialize ~~to~~ the object.

c.) Copy Constructors

A copy constructor is a member function which intializes an object using another object of the same class. Whenever we define one or more non-default constructors for a class, a default constructor Should also be explicity defined as the Compiler will not provide default constructor in this case.

3. Explain the difference b/w object oriented and procedural programming language in detail.

s ⇒ Object oriented programming

It can be defined as a programming model which is based upon the concept of object. object contain data in the form of attributes and code in the form of methods. In object oriented programming, computer programms are designed using the concept of object that interact with real world. Object oriented programming language are various but the most popular one are class based, meaning that objects are instance of classes, which also determine their type. Example of object oriented programming languages are Java, C++, C#, Python, Ruby.

Procedural language

It can be defined as a programming model which is derived from Structured programming, based upon the concept of calling procedure. Procedure also known as a routine, Subroutine or functions, Simply consist of a series of computation steps to be carried out. During programs execution any given procedure might be called at any point, including by other procedures or itself. Example of procedural languages are fortan, Pascal and C.

long Answer Type Question

A.) Explain the type of polymorphism with code.

Ans ⇒ In C++ polymorphism is mainly divided into twotypes:

1.) Compile time polymorphism

This is type of polymorphism is achived by function overloading or operator overloading.

```
# include <bits/std c++.h>
  using namespace std;
    Class Poly
    { public:
      void func (int n)
      { Cout << "value of n is" << n<< endl;
        }
      void func (double x)
      {  cout << "value of n is << x << endl;
       }
      void func (int n, int y)
      {  Cout << "value of n and y is" <<n<< "," << y<< end;
        }
          };
     int main () {
       Greeks ~~obj 1~~ Poly 1;
         Poly 1. fuc (7);
         Poly 1. func (9.132);
         poly 1. fuc (85, 64);
          return 0;
         }
```

2.) Runtime Polymorphism:
This type of polymorphism is achived by function Overriding.

```
# include <bits/std c++. 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" << endl;}
         void Show ()
         { Cout <<"Show derived class" << endl; }
         };
         int main ()
         { base *bptr;
            derived d;
            bptr = &d
            bptr -> pirt();
             bptr -> show ();
             return 0;
              }
```

B.) Write a program to sort an array of 0,1,2 in the best time and space complexity.

Ans:->
```
# include <bits/stdc++.h>
 using namespace std;
  void Sort 012 (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;
           }
             } }

   void print array (int arr[]; int arr-sizes)
    {   for (int i = 0; i < arr-size; i++)
            cout << arr[i] << " ";
        }
```

```
int main ()
{ int arr[] = {1 1 2 2 0 0 2 1 2};
  int n = size of (arr) / size of (arr[0]);
  Sort 012 (arr, n);
  Cout << "array after Segragation";
  Print array (arr, n);
  return 0;
  }
```

C.) Create a class named 'member' having the following members:
Data members
1. Name
2. Age
3. Phone number
4. Address
5. Salary.

Ans :=
```
# Include <bits/Stdc++.h>
using namespace Std;
class member {
 char name [20] , address[40];
   double number;
   int age ;
   public :
    intSalary ;
    void input ()
```

```
{ Cout <<endl;
  Cout <<"Name:<< endl;
  Cin. get line (name,20);
   Cout <<"Age : "<<endl ;
    Cin >> age;
     Cout<< "phone Number:"<<endl";
       Cin >> number
       Cout<< " Address: "<< endl;
        Cin. getline (address, 40);
        cout << Salary ;"<< endl';
          cin >> Salary ;

      }
       void display()
      { Cout << endl;
        cout<< " Name:"<< name << endl;
        Cout << " Age : " << Age << endl;
        cout << " Phone Number :"<< number<< endl;
        Cout << " Address: "<< address << endl;
        Cout << " Salary: " << Salary << endl;

       } };
     Class employee : public member {
       Char specialization[20], department [20];
     public:
     void iput()
     { Cout<<" \n\t enter Employee Detail \t\n";
```