Date: / /
Page No.

I. Choose the correct option:-

Q1 Local varitables are started in an area called Stack

(a) Heap (c) Free memory
(b) permanent (d) stack

Q2 Choose the correct option?

```
# include using
namespace std;
class Base {};
class Derived : Public
Base {}
int main()
{
Base *bp =new
Derived,
Derived * dp=new
Base ; }
```

(a) No Compiler Error
(b) Compiler Error in line Base
(c) Compiler Error in line Derived * dp= new Base
(d) Runtime Error

Q3 When the interitance is private, the private methods in base class are Inacceside in the derived class (in C++)

(a) In acceside (c) Protected
(b) Accessible (d) public

Date: / /
Page No.

Q4 which of the following is true ?
(a) The number of times destracter is called depends on number of objects created.
(b) Destructer is called only once
(c) there can be more than one destructors at the class.
(d) programmer have always call destructor at the end of the program.

Q5 State true or False
(a) Type Conversion is automatic whereas type casting is explict
(a) True
(b) False

II. Short Answer type question:-

Q1 Explain about new and delete keyboards of code
Ans: syntax to use new operator: To allocate memory of any data type, the syntax is:
pointer-variable= new data-type;

delete operator is use to deallocate the memory user privilege to deallocate the created pointer variable by this delete operator

Syntax:- // Release memory allocation pointed by pointer variable
delete pointer-variable

Date : / /
Page No.

Lets topic a example:-
Allocation and deallocation of memory using new and delete

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int *p= NULL;
p= new (nothrow) int;
if (!P)
cout << " allocation of the memory failed \n";
else
 * p= 89;
cout << " value of p: " << * p << endl;
}
Float *r = new float (12.86);
cout << " value of r: " << *r << endl;
int n=8;
int *q = new (nothrow )int [n];
if (!q)
cout << allocation on the memory failed in ";
else {
      For (int i =0; =1<n; i++)
        q [i] = i+1;
cout << value store in block of memory: ";
For (int i =0; i<n; i++)
cout <<q[i] << " " ; }
```

v & delete operator:-

Go Run Terminal Help new_delete.cpp - c++lpu - Visual Studio Code

Getting Started | hello.cpp | new_delete.cpp | default.cpp | poly.cpp | sort.cpp | class.cpp | variables.cpp

assignment2 > new_delete.cpp > main()

```cpp
#include <bits/stdc++.h>
using namespace std;

int main ()
{
    int* p = NULL;
    p = new(nothrow) int;
    if (!p)
        cout << "allocation of memory failed\n";
    else
    {
        *p = 89;
        cout << "Value of p: " << *p << endl;
    }
    float *r = new float(12.86);

    cout << "Value of r: " << *r << endl;
    int n = 8;
    int *q = new(nothrow) int[n];

    if (!q)
        cout << "allocation of memory failed\n";
    else
    {
        for (int i = 0; i < n; i++)
            q[i] = i+1;

        cout << "Value store in block of memory: ";
        for (int i = 0; i < n; i++)
            cout << q[i] << " ";
    }
    delete p;
    delete r;
    delete[] q;

    return 0;
}
```

Command Prompt

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ new_delete.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
Value of p: 89
Value of r: 12.86
Value store in block of memory: 1 2 3 4 5 6 7 8
C:\Users\hp\Desktop\c++lpu\assignment2>
```

Ln 10, Col 9 Tab Size: 4

Date : / /
Page No.

```
delete p;
delete r;
delete [] q;
} return 0;
```

OP :- value of p = 89
value of r = 12:89
value store in block of memory:
12345678

2 what are constructors? why they are required? Explain different types of constructors with suitable example.

ns A constructor is a member Function of a class with intlli object of a class. In C++ constructor is automatically called when object (istance of class) create. It is special member function of class. The main purpose of the class constructor in C++ programing is to construct an object of a class. In other word it is used to intilidge all class data members.

Types of constructors:-

1. Default constructor:- Default constructor is the construct which doesn't take any arguement It has no paramete

Syntax:- class name (parameter1, parameter2 .....)
{ // constructor Definition
}

example:-

```
#include <bits/stdc++.h>
using newspaper std;
class cube
{
public
int side;
```

Date : / /
Page No.

```
cube c)
{ side = 10;
} };
int main c)
{
 Cube C;
 Cout << C side;
 } return 0; }
 O/P → 10
```

2. parameterized constructor :- These are the constructors with parameter using this contructor you can provide different values to data members of different objects by passing the appropriate values as argument.

example :-

```
# include <bits/stdc++.h>
using namespace std;
  class cube
  { public
  int side;
    cube (int x)
    {
      side = x;
  } };
 int main c)
{
  Cube C1 (10);
```

Date : / /
Page No.

```
cube c2 (20);
cube c3 (30);
~~cube c4 (40)~~;
cout << c1 side;
cout << c2 side;
cout << c3 side;
return 0;
}
```

output :- 10
20
30

(3) Copy constructer:- It is used in create a copy of an already exisiting object of class type. It is usually of the form x(x&), where x is the class name. The compiler provides a default copy constructer to all the classes.

Syntax:-

```
class name (const class name & object name)
{.....
```

example:-

```
# include <bits/stdc++.h>
using namespace std;
class Sample Copy Constructor
{
private:
int x, y;
public:
Sample Copy constructor (int x1, int y1)
```

//default constructor:-

```
#include<bits/stdc++.h>
using namespace std;
class cube
{
    public:
    int side;
    cube()
    {
        side=10;
    }
    };
    int main()
    {
        cube c;
        cout<<c.side;

        return 0;
    }
```

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ default.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
10
C:\Users\hp\Desktop\c++lpu\assignment2>
```

//parameterized constructor:-

```
#include<bits/stdc++.h>
using namespace std;
class Cube
{
    public:
    int side;
    Cube(int x)
    {
        side=x;
    }
};

int main()
{
    Cube c1(10);
    Cube c2(20);
    Cube c3(30);
    cout << c1.side <<endl;
    cout << c2.side <<endl;
    cout << c3.side <<endl;
    return 0;
}
```

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ para.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
10
20
30

C:\Users\hp\Desktop\c++lpu\assignment2>
```

//copy constructor:-

copy.cpp - c++lpu - Visual Studio Code

Getting Started | hello.cpp | new_delete.cpp | default.cpp | para.cpp | copy.cpp | sort.cpp | class.cpp

assignment2 > copy.cpp > _

```cpp
#include<bits/stdc++.h>
using namespace std;
class Samplecopyconstructor
{
    private:
    int x, y;
     public:
    Samplecopyconstructor(int x1, int y1)
    {
        x = x1;
        y = y1;
    }
    Samplecopyconstructor (const Samplecopyconstructor &sam)
    {
        x = sam.x;
        y = sam.y;
    }
     void display()
    {
        cout<<x<<" "<<y<<endl;
    }
};
int main()
{
    Samplecopyconstructor obj1(10, 15);     // Normal constructor
    Samplecopyconstructor obj2 = obj1;      // Copy constructor
    cout<<"Normal constructor : ";
    obj1.display();
    cout<<"Copy constructor : ";
    obj2.display();
    return 0;
}
```

Command Prompt

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ copy.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
Normal constructor : 10 15
Copy constructor : 10 15

C:\Users\hp\Desktop\c++lpu\assignment2>
```

Ln 22, Col 3

ere to search

Date : / /
Page No.

```
{
  x=x1;
  y=y1;
}
Sample copy constructor (const sample &place-copy construct
{
x= Sam.x;
y= Sam.y;
}
void display ()
{
cout<<x<<"  "<<y <<endl;
} };
int main ()
{
Sample copy constructor obj1 (10, 15);
Sample Copy Constructor ob2 (=obj1;
cout <<"Normal Constructor:";
obj1.display ();
cout <<" Copy Constructor:";
obj 2. display ();
return0;
}
```

output:-
copy constructor: 10, 15

Date : / /
Page No.

Q3 Explain the difference between object oriented and procedural programming language is detail.

Ans →

object-oriented programing	procedural oriented programming
(i) In oop program is divided into small parts called object	(i) In procedual programming, program is divided into small parts called functions.
(ii) It follows bottom up approach	(ii) It follows top-down approach
(iii) It can access specifier like public, private and protected	(iii) There is no access specifies in procedural programming
object oriented programing	procedural programming
(iv) Adding new data and fuction is easy	(iv) Adding new data and function is not easy.
(v) object oriented programming provides data hiding & it is more secure.	(v) procudural programming does not have any proper way for hiding data & it is less.
(vi) overloading is possible in object oriented programming	(vi) In this programming overloading is not possible.
(vii) In object oriented programming data is more important than function.	(vii) In procedural programming function is more important than data
(viii) It is based on real world	(viii) It is based on unreal world

Long Answer type question:-

Explain the type of polymorphis with code:-

poly morphism :- The word polymorphism means having many forms. poly morphism means that a call to a member function will cause a different function to be executed depending on the type of object invoices the function

Type of polymorphism

Compile time polymorphism Run time polymorphism

Compile time polymorphism :- This is also known as static Corearl binding function overloading and operators overloading are perfect example of compile time polymorphism.

Example :- In this example we have two functions with sa name but different number of arguments Based On how many parameter we pass during functions calls determi which function is to be called this is why it is conside as an example of polymorphism because in different condit the output is different since the call is determined durin compile time that's why it is called compile time polymorp

Code :- # Include <bits/stdc++h>

Using namespace std;

class Add {

//compile time polymorphism:-

assignment2 > compile.cpp

```cpp
#include <bits/stdc++.h>
using namespace std;
class Add {
public:
  int sum(int num1, int num2){
    return num1+num2;
  }
  int sum(int num1, int num2, int num3){
    return num1+num2+num3;
  }
};
int main() {
  Add obj;
  cout<<"Output: "<<obj.sum(10, 20)<<endl;
  cout<<"Output: "<<obj.sum(11, 22, 33);
  return 0;
}
```

Select Command Prompt

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ compile.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
Output: 30
Output: 66
C:\Users\hp\Desktop\c++lpu\assignment2>
```

Date: / /
Page No.

```
Public:
int sum (int num1, int num2){
return num1+ num2;
}

int sum(int num1, int num2, int num3){
  return num1+num2+ num3;
} };

int main(){
Add obj;
cout << "output: " << obj.sum (10, 20) <<endl;
cout << " output: " << obj sum (11, 22, 33);
return0;
}
```

O/P:- output : 30
output : 66

(ii) Run time polymorphism :- This is also known as dynamic (or late) binding

Example:- Function overriding is an example of Runtime polymorphism.

Function overriding: when child class declares a method which is already present in the parent class then their is called function overriding here child class overrides the parent class. In case of function overriding we have two definitions of same function one is parent class and one is child class. The call the function is determined at runtime to decide which definition of the function.

Code :-
```
#include <bits/stdc++.h>
Using namespace std;
Class A {
Public :
Void display () {
Cout << "Super class Function" <<endl;
};
Class B : Public A {
Public :
Void disp() {
Cout <<"Subclass function";
};
int main () {
A obj;
Obj. disp();
B obj2;
obj2. disp();
return 0;
}
```

O/P :- Superclass function
Subclass Function

//runtime polymorphism:-

runtime.cpp - c++lpu - Visual Studio Code

Getting Started | hello.cpp | new_delete.cpp | default.cpp | para.cpp | copy.cpp | compile.cpp | runtime.cpp

assignment2 > runtime.cpp > ...

```cpp
#include <bits/stdc++.h>
using namespace std;
class A {
public:
  void disp(){
      cout<<"Super Class Function"<<endl;
  }
};
class B: public A{
public:
  void disp(){
      cout<<"Sub Class Function";
  }
};
int main() {

  A obj;
  obj.disp();
  B obj2;
  obj2.disp();
  return 0;
}
```

Command Prompt

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ runtime.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
Super Class Function
Sub Class Function
C:\Users\hp\Desktop\c++lpu\assignment2>
```

Ln 1, Col 24 Spaces

Q2) Write a program to sort an array of 0, 1, 2 in the best possible time and space complexity.
For example :-
Input Array :- 112200212
Output Array :- 001112222

Ans→ Code :-
```
#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_size)
{
  for (int i=0; i<arr_size; i++)
  Cout <<arr[i]<<" ";
}
```

// array after segmentation:-

sort.cpp - c++lpu - Visual Studio Code

Getting Started | hello.cpp | new_delete.cpp | default.cpp | para.cpp | copy.cpp | compile.cpp | runtime.cpp

assignment2 > sort.cpp > sort012(int [], int)

```cpp
#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;
        }
    }
}
void printArray(int arr[], int arr_size)
{
    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 = sizeof(arr) / sizeof(arr[0]);
sort012(arr, n);
cout << "array after segregation : ";
printArray(arr, n);
return 0;
    }
```

Command Prompt

```
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Desktop\c++lpu\assignment2

C:\Users\hp\Desktop\c++lpu\assignment2>g++ sort.cpp

C:\Users\hp\Desktop\c++lpu\assignment2>a
array after segregation : 0 0 1 1 1 2 2 2 2
C:\Users\hp\Desktop\c++lpu\assignment2>
```

Ln 13, Col 1 Tab Size:

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

o/p → array after segregation: 0 0 1 1 1 2 2 2 2

Q3) Create a class named 'Member' having the following members:

Data members

1- Name
2- Age
3- Phone number
4- Address
5- Salary

It also has a method named 'Print Salary' which prints the salary of members. Two classes 'Employee' and 'manager' inherits the 'member' class. The 'Employee' and 'manager' classes have data members 'specilization' and 'department' respectively. Now assign name, age, Phone no., address and salary to an employee and a manager by making an object of both these classes and print the same.

ns→ Code :-

```
#include <bits/stdc++.h>
Using namespace std;
```

```
Class member {
Char name [40], address [50];
int number;
int age;
Public:
int salary;
Void input ()
{
Cout << "Name : " << endl;
Cin >> name;
Cout << "Age : " << endl;
Cin >> age;
Cout << "Phone number : " << endl;
Cin >> number;
Cout << "Address : " << endl;
Cin >> address;
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 [30], department [20];
Public :
Void input ()
{
 Cout <<"\n Enter employee Details:";
  member :: input ();
Cout <<"specilization :" <<endl;
 Cin >> Specilization ;
 Cout <<"Department :" <<endl;
  Cin >> department ;
}
Void display ()
{
Cout <<"\n Displaying Employee Details \n";
 member :: display ();
Cout <<"Specialization :" << specialization<<endl;
Cout <<"Department : "<< department <<endl;
}
 Void Print Salary ()
{
 Cout <<"\n salary of the member is : "<< salary<<end
}};
Class manager : Public member {
char Specialization [30],department [20];
Public :
Cout <<"\n Enter manager Details \n";
 member :: input ();
 ~~Cin >>~~ Cout <<" Specialization : "<< endl;
          Cin >> specialization;
```

```
Cout << "Department : " <<endl;
 Cin >> department ;
}
 Void display ()
{
Cout << "\n Displaying manager details \n";
member :: display ();
Cout << " Specialization : "<<specialization <<endl;
Cout << "Department : " << department <<endl;
}
 Void PrintSalary ( )
{
Cout << "\n Salary of the member is : " <<salary <<end
}};
int main ()
{
 employee e;
 manager m;
 e. input ();
 e. display ();
 e. printsalary ();
  m. display ();
  m. printsalary ();
 return 0;
}
```

O/P :- Enter Employee Details :

Name : John

Age : 21

Phone Number : 0123456789

Address : Punjab
Salary : 12000000
Specialization : developer
Department : CSE
Enter Manager Details :
Name : Laraib
Age : 28
Phone Number : 012345678
Address : Bgp
Salary : 1280000
Specialization : Coder
Department : CSE
Displaying Employee Details :
Name : John
Age : 21
Phone number : 0123456789
Address : Punjab
Salary : 12000000
Specialization : developer
Department : CSE
Salary of the member is : 12000000

Displaying Manager Details :
Name : Laraib
Age : 28
Phone Number : 012345678
Address : Bgp
Salary : 1280000
Specialization : Coder
Department : CSE
Salary of the member is : 1280000

//inheritance:-

```
#include <bits/stdc++.h>
using namespace std;
class member{
  char name[40], address[50];
   int number;
   int age;
   public:
    int salary;
    void input()
    {   cout<<endl;
     cout<<"Name : "<<endl;
     cin>>name;
     cout<<"Age : "<<endl;
     cin>>age;
     cout<<"Phone Number : "<<endl;
     cin>>number;
     cout<<"Address : "<<endl;
     cin>>address;
     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[30], department[20];
 public:
 void input()
 {
 cout<<"\n Enter Employee Details: ";
 member::input();
 cout<<"Specialization : "<<endl;
 cin>>specialization;
```

```
 cout<<"Department : "<<endl;
 cin>>department;
 }
 void display()
 {
 cout<<"\n Displaying Employee Details \n";
 member::display();
 cout<<"Specialization : "<<specialization<<endl;
 cout<<"Department : "<<department<<endl;
 }
 void printSalary()
    {
    cout<<"\n Salary of the member is : "<<salary<<endl;
 }

};
class manager : public member{
 char specialization[50], department[20];
 public:
  void input()
 {
 cout<<"\n \t Enter Manager Details \t \n";
 member::input();
 cout<<"Specialization : "<<endl;
 cin>>specialization;
 cout<<"Department : "<<endl;
 cin>>department;
 }
 void display()
 {
 cout<<"\n \t Displaying Manager Details \t \n";
 member::display();
 cout<<"Specialization : "<<specialization<<endl;
 cout<<"Department : "<<department<<endl;
 }
 void printSalary()
```

```
    {
    cout<<"\n \t Enter Manager Details \t \n";
    member::input();
    cout<<"Specialization : "<<endl;
    cin>>specialization;
    cout<<"Department : "<<endl;
    cin>>department;
    }
    void display()
    {
    cout<<"\n \t Displaying Manager Details \t \n";
    member::display();
    cout<<"Specialization : "<<specialization<<endl;
    cout<<"Department : "<<department<<endl;
    }
    void printSalary()
      {
      cout<<"\n Salary of the member is : "<<salary<<endl;
    }
};
int main()
{
 employee e;
 manager m;
 e.input();
 m.input();
 e.display();
 e.printSalary();
 m.display();
 m.printSalary();

 return 0;
}
```

```
Command Prompt
1200000
Specialization :
coder
Department :
cse

 Displaying Employee Details

Name : John
Age : 21
Phone Number : 123456789
Address : punjab
Salary : 12000000
Specialization : developer
Department : cse

 Salary of the member is : 12000000

        Displaying Manager Details

Name : Jaresh
Age : 20
Phone Number : 1234567890
Address : hpp
Salary : 1200000
Specialization : coder
Department : cse

 Salary of the member is : 1200000
```