ALTAF SIDDIQUE

Assignment - 2

* Choose the Correct option:-

1 Q. D

2 Q. C

3 Q. A

4 Q. C

5 Q. TRUE

* Short Answer Type Questions:-

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

New operator is used for dynam memory allocation which puts variabl on heap memory.

```
class Car {
String name;
int num;
Public:
Car (string a, int n) {
Cout<<" Constructor Called "<< endl;
name = a.
}
 void enter()
 {
   Cin >> name;
   Cin >> num;
void display()
{
  Cout << "Name: "<<name<<endl;
  Cout<<"Num: <<num<<endl;
}
};
  int main()
  {
    Car*P = new Car ("Honda", 2021);
    P-> display()
  }
```

2Q) A Constructor is a special type of function with no return type. Name of Constructor should Be Same as the name of Class.

Constructor Types:-

1. Default Constructor
2. Parameterized Constructor
3. Copy Constructor
4. Static Constructor
5. Private Constructor

3Q)

	POP	OOP
i.	Program is divided in small parts called function	i. Program is divided in small part called object
ii.	Procedural Programming follow top down approach	ii. object Oriented programming follows bottom up approac
iii.	There is no access Specifier in Procedural Programming.	iii. OOP has access Speci like Private, Public, Protected etc.
iv.	Adding new data & function is not easy.	iv. Adding new data & function is easy

bonafide.png

CLASSMATE
Date :
Page :

★ Long Answer Type Questions:-

4 Al Polymorphism is mainly divided into two types
= ① Compile Time Polymorphism
② Runtime Polymorphism.

① C.T.P — This Type of Polymorphism is achived by function overloading or operator overloading.
When there are Multiple functions with same name but different Parameters then these function are Said to Be overloaded.

Ex —
```
Class Complex {
Private:
int real, imag;
Public:
Complex(int r = 0, int i=0){real=r, img=i;}
Complex operator + (Complex Const &obj)
{
 Complex res;
 res. real = real + obj. real;
 res. img = imag + obj. imag;
 return res;
}
 Void print()
{
 cout<< real <<"+i"<<imag<< real<<endl;}
};
```

CLASSMATE
Date :
Page :

= Runtime Polymorphism :- This type of Polymorphism is achived By function overriding.
when a derived class has a defination for one of the member functions of the base class. that Base function is said to be overridden

Ex -
```
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 -> print()
bptr -> show();
return 0;
}.
```

B Q1

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

⟹ Output

```
0 0 0 0 0 1 1 1 1 1 2 2
```

Q8
=
```
class Member {
char name [20]  address [40];
double number;
int age;
public:
int Salary;
void input() {
cout<<endl;
Cout<< "Name: "<<endl;
Cin.getline (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<<e
Cout<<"Address:" <<address <<endl;
Cout<<"Salary:"<<Salary << end;
}
};
```

```
class employee: Public Member {
void printSalary ()
{
Cout<<"\n Salary of the member is: <<Salary
}
};
int main()
{
m. display ()
m. printSalary():
}
```