Question # 01 **[20+10+10=40 Points]**

A. Find out logical & syntax error. For each error, mention either logical or syntax & correct the error without any reasoning: (10+10=20 Points)

```
void print1(int x[], const int SIZE){
   ...
}
void print2(int x[][5], const int NROWS){
   ...
}
void print3(int **x, const int SIZE){
   ...
}
int main(){
   print1 (b , 15);
   print1 (a , 15);
   print1 (b[0] , 15);
   print2 (a , 2);
   print3 (b , 15);
   print3 ((int**)b , 15);
```

//Syntax Error- b is 2D, where function has first parameter 1D therefore, either write b[0] or &b[0][0] or type cast (int*)

//Logical Error – replace 15 with 10

//No Error – 2D array is passed as 1D array, which is possible

//Syntax Error – a is 1D, where function has first parameter 2D therefore type cast array a (int (&)[ ][5]) to pass to function

//Syntax Error – b is 2D static array, where function has first parameter ** dynamic pointer, the solution is to type cast array into (int), though it will be a logical error, because b has only 1 address, where a 2D pointer has multiple addresses, minimum 2**

//Logical Error– b can't be passed to print 3 because it is a 2D static //array, have only one address can't work as 2D dynamic array

---

```
int g=5;
class A{
   int x;
public:
   int y=5;
   A& A(){   x=g;  }
   A(const A a){   ...  }
   A operator *= (const A&);
   A& operator / (const A& a){
      A newA;
      ...
      return newA;
   }
};
```

// Syntax error – we can't initialize data member like this

//Syntax error – constructor can't have return type

//Syntax error – Copy constructor should have parameter as reference

//Logical error - *= is supposed to modify current object and to return current object return type should be A&

//Logical error, returning local object, which should be returned by creating a copy of local object

B. Suppose your program contains the following class definition and suppose the main function of your program contains the following declaration: (10 Points)

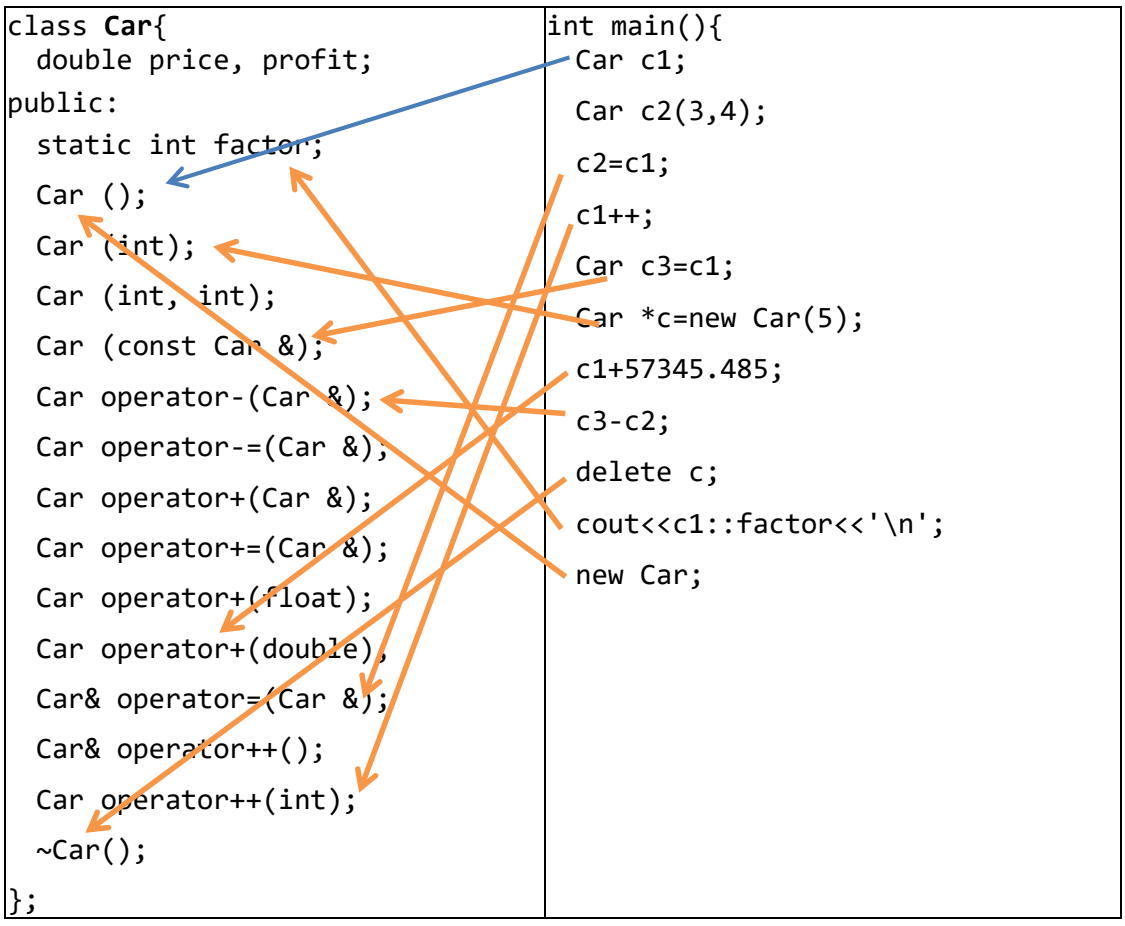
```
Car hyundai(500...), jaguar(700...), cultus(600...);
```

Which of the following statements are allowed in the main function of your program? (For legal statements write 'Y' and 'N' for illegal?

```
class Car{
  double price;
  double profit;
  double getProfit();
public:
  Car (int);
  Car (const Car &);
  void setPrice(double);
  void setProfit(double);
  double getPrice();
  double operator-(Car &);
  Car operator+(Car &);
};
```

Statements		Legal [Y/N]
`jaguar.setPrice(30.97);`		Y
`jaguar.getPrice();`		Y
`jaguar.getProfit();`	**Accessing private member function**	N
`cout<<"Price:"<<price<<'\n';`	**Accessing private data member**	N
`hyundai.setPrice(jaguar.getPrice());`		Y
`hyundai = jaguar;`		Y
`Car *p=new Car;`	**Non-parameterized constructor does not exist**	N
`float diff = jaguar - hyundai;`		Y
`hyundai.price = 4999.99;`	**Accessing private data member**	N
`Car newC=cultus + hyundai -jaguar;`	**Binary operator – has return type double, where we are storing result in a new object, which means trying to call copy constructor, where copy constructor requires object of type Car**	N

C. Consider code. For each statement in main point an arrow towards relevant member Function: (10 Points)



Question # 02 **[5+15+5+20=45 Points]**

Write code/functions for the following descriptions.

A. Write global function "**add**" to add 2 objects of Point class and return a new object of Point class. **[3]**

```
Point add (const Point &p1, const Point &p2){
        Point p3 = p1;
        p3.x += p2.x;
        p3.y += p2.y;
        return p3;
}
```

```
class Point {
    int x, y;
public:
  //constructors & getter
  //setter given here
};
```

B. Provide the implementation of the following class named **Fuzzy**. Each **data member** of this class should contain **2, 1** or **0 (default value is 0)** for a particular object. **[3+3+4+2+3=15 Points]**

```
class Fuzzy{
        int x;
        int y;
```

//Write one set function to set value of both data members

```
void set (int x, int y){
      if (x<0 || x>2) x = 0;
      if (y<0 || y>2) x = 0;
      this -> x = x;
      this -> y = y;
}
```

//Write a non-parameterized constructor to set default value of DM's, call set function inside

```
Fuzzy (){
      set(0, 0);
}
```

//overload minus (–) binary operator to subtract two two objects by subtracting their data members and return the result in new object, remember value of DM's should remain legal, otherwise assign default value

```
Fuzzy& operator - (const Fuzzy &f){
      Fuzzy temp = *this;
      temp.x = x – f.x;
      temp.y = y – f.y;
      if (temp.x < 0) temp.x = 0;
      if (temp.y < 0) temp.y = 0;
      return temp;
}
```

//Write static member function to print information about class "This is a fuzzy class"

```
static void about(){
      cout << "This is a fuzzy class\n";
}
```

//Write post increment operator ++ to add 1 into both data members, if value is less than 2, otherwise don't add anything

```
Fuzzy& operator++ (int){
      Fuzzy temp = *this;
      if (x<2) x++;
      return temp;
}
```

C. Consider class data members and output (inside the box), write member function show? **[5]**

```
class ABC{
      int age;//for less than 10 print Invalid in show function
      char status; //E for Enable, D for Disable
      int count; //Count of values in x1 & x2
      int *x1, *x2;
      void show() const{
            cout << "Age: " << age << "years\n";
            cout << "Status: ";
            if (status == E)  cout << "Enable\t";
            else              cout << "Disable\t";
            cout << "Count: " << count << '\n';
            cout << "\tX1\tX2\n--------  --------\n";
            for (int i=0;i<count;i++){
                  cout << right << setw(5) << x1[i] << '\t';
                  cout << setw(5) << x2[i] << '\n';
            }
      }
```

```
Age: 12 years
Status: Enable   Count: 3
   X1       X2
---------- ----------
   23       18
  126       29
   34      108
```

D. Define following briefly: **[20]**

a. Typical getter function for a data member named **center** of type **float** in a class **Line**.

```
float getCenter() const{
      return center;
}
```

b. Typical setter function for a data member of **Set** class named **size** of type **int**. Setter function must implement a check on **size** as non-negative.

```
void setSize(int size) {
      if (size<0) size=1;
      this->size = size;
}
```

c. Definition of a destructor of **File** class to close the file pointed by a ***data member*** *(*ofstream objects*)* named outputFile

```
~File(){
      outputFile.close();
}
```

d. Two different prototypes declarations (only not definition) of the Copy constructor for a class named **Cashmemo**

```
Cashmemo(Cashmemo & cm);
Cashmemo(const Cashmemo & cm);
```

e. Write the definition of the explicit operator **!=** for class named **RationalNumber** composed two integers ***num*** and ***den***. An example rational number is at right whose **double** equivalent is ***num/den***.

```
bool operator !=(const RationalNumber &r){
      return (double)num/den == (double)r.num/r.den;
}
```

f. Write definition for the overloading operator **<=** as a member **ComplexNumber** class. Two complex numbers have to be compared through their modulus ($\sqrt{x^2 + y^2}$).

```
bool operator <=(const ComplexNumber &c){
      double result1 = pow(x*x + y*y, 0.5); //or sqrt (x*x + y*y)
      double result2 = pow(c.x*c.x + c.y*c.y, 0.5); //or sqrt (c.x*c.x + c.y*c.y);
      return result1 <= result2;
}
```

g. Complete following code to save object **obj** in both files.

```
int main(){
    ofstream out1("data.txt"); //text file
```

```
    ofstream out2("data.bin", ios::binary);
    Simple obj(...);
  out1 << obj.x << ' ' << obj.f << ' ' << obj.d << '\n';
  out2.write ( (char*) &obj, sizeof (Simple));
```

h. If member function perfectly **copy** (making deep copy) for class **Klacc** is available, write the definition of **copy constructor** & **assignment operator** using **copy** function?

```
Klacc (const Klacc &k){
  copy(k);
}
Klacc& operator = (const Klacc &k){
  delete []c;
  for (int i=0;i<count;i++)
      delete []d[i];
  delete []d;
  return copy(k);
}
```

```
class Klacc{
  //Use for dynamic arrays
   int *c; // for 1d
   float **d; // for 2d
};
```

i. Give an example with necessary code of any constructor of class named **AAA**, to initialize its *const data member* ***CC*** of type **double** with the value provided as a parameter.

```
AAA (...,double cc) : CC(cc){...}
```