ASSIGNMENT

Question.1

Demonstrate the concept of "Has a" relationship: You are required to implement following class diagram (complete program) in C++ to demonstrate the concept of Composition(Website is composed of webpages and links)with all data members, constructors, member functions . Add at least 4 webpages objects in a website class. And each webpage should have at least two hyperlinks.

INPUT:

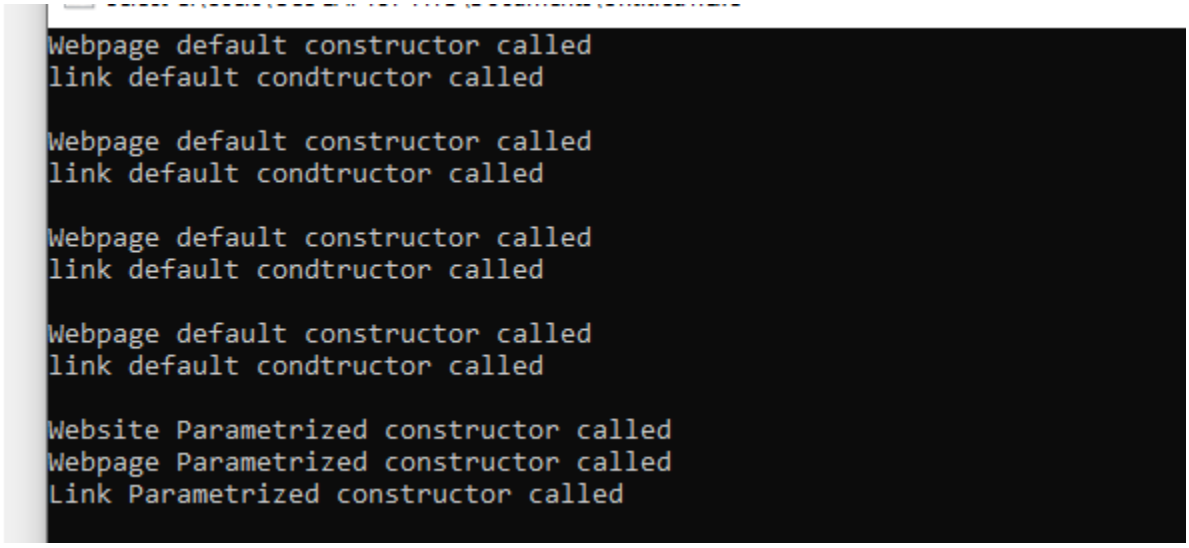
```
#include <iostream>
#include <conio.h>
#include <string.h>
using namespace std;
class Webpage
{
 double width;
 double height;
public:
 Webpage()
 {
 cout << "Webpage default constructor called" << endl;
 };
 Webpage(double w, double h)
 {
 width=w;
 height=h;
 cout << "Webpage Parametrized constructor called" << endl;
 };
};
```

```
class Link
{
 char name;
public:
 Link()
 {
 cout << "link default condtructor called\n"
 << endl;
 }
 Link(char network)
 {
 name=network;
 cout << "Link Parametrized constructor called" << endl;
 };
};
class Website
{
 char name;

public:
 Website()
 {
 cout << "Website constructor Called" << endl;
 };
 Website(char amazon)
 {
 name=amazon;
 cout << "Website Parametrized constructor called" << endl;
 };
```

```
 Webpage k4;
 Link i4;
 Webpage k3;
 Link i3;
 Webpage k2;
 Link l2;
 Webpage k1;
 Link l1;
};
int main()
{
 Website www('s');
 Webpage wp(45.5, 15.9);
 Link i('n');
 return 0;
}
```

OUTPUT:



Question .2

Design a class named Fan to represent a fan. The class contains:

• Three constants named SLOW, MEDIUM and FAST with value 1, 2, and 3 to denote the fan speed . • An int data field named speed that specifies the speed of the fan (default SLOW).

• A boolean data field named on that specifies whether the fan is o (default false)

• A double data field named radius that specifies the radius of the fan (default 5).

• A string data field named color that specifies the color of the fan (default blue).

• A default constructor that creates a default fan.

• The accessor and mutator methods for all four data fileds.

• The method named Showdata() that returns a string description for the fan. If the fan is on, the method returns the fan speed, color and radius in one combined string. If the fan is not on, the method returns fan color and radius along with the string “fan is off” on one combined string

INPUT:

```
#include <iostream>
#include <conio.h>
#include <string.h>
using namespace std;
class fan
{
 const int slow = 1 ;
 const int medium =2 ;
 const int fast =3;
 int speed;
 bool on;
 double radius;
```

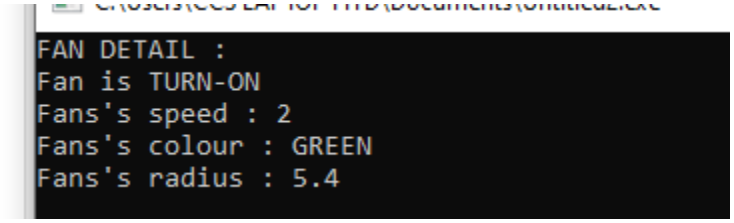
```
string colour;
public:
        fan(){

        };
fan(int s,bool c,double t,string we)
{
speed = s;
on = c;
radius = t;
colour = we ;
};
int getspeed()
{ return speed;
}
void setspeed(int s)
{
speed = s;
}
bool geton()
{
return on;
}
void seton(bool o)
```

```
{
on = o;
}
double getradius()
{
return speed;
}
void setradius(double r)
{
radius = r;
}
string getcolour()
{
return colour;
}
void setcolour(string c)
{
colour = c;
}
string showData()
{
if (on == true)
{
cout << "FAN DETAIL : " << endl;
```

```
 cout << "Fan is TURN-ON " << endl;
 cout << "Fans's speed : " << speed << endl;
 cout << "Fans's colour : " << colour << endl;
 cout << "Fans's radius : " << radius << endl;
 }
 else
 {
 cout << "FAN DETAIL : " << endl;
 cout << "Fan is  TURN-OFF " << endl;
 cout << "Fans's colour : " << colour << endl;
 cout << "Fans's radius : " << radius << endl;
 }
 }
};
int main(){
fan G(2,true,5.4,"GREEN");
G.showData();
 return 0;
}
```

OUTPUT:



Question .3

A class RentedVehicle that has:
• One private instance variable baseFee of type double
• One constructor to initialize the instance variable
• One instance method getCost () that returns the base fee –
Accessor methods for the instance variables
A subclass FuelVehicle that :
• has one additional private instance variable Kms indicating the total number of kilometers traveled.
• one constructor to initialize the instance variables.
• one instance method getMileageFees to return the fees due to mileage based on the following:
If Kms < 100 mileagefees=0.2*kms
If 100<=Kms<= 400 mileagefees=0.3*kms
If Kms>400 mileagefees=0.3 times 400 plus 0.5 times the extra kilometers above 400. –
accessor methods
A Car class which is a subclass of FuelVehicle that :
• has one additional private instance variable Seats
• has one constructor to initialize the instance variables
• overrides getCost method by adding seats*baseFee to mileageFees
• accessors
A Truck class which is a subclass of FuelVehicle that:
• has one private instance variable capacity
• has one constructor to initialize the instance variables
• overrides getCost method by adding baseFee*capacity to mileageFees
• accessors
A Bicycle class that extends RentedVehicle that:
• has one additional private instance variable nDays indicating the number of days it is rented.
• has one constructor to initialize the instance variables
• overrides getCost method to return baseFee * nDays
accessors
Implement all five classes with their accessor and mutator methods.

INPUT:

```
#include <iostream>

#include <conio.h>

#include <string.h>

using namespace std;

class rentedVehicle

{

private:
```

```
 double baseFee;
public:
 rentedVehicle()
 {
 };
 double getCost()
 {
 return baseFee;
 }
 void setCost(double bond)
 {
 baseFee = bond;
 }
};
class FuelVehicle : public rentedVehicle
{
 int kms;
public:
 FuelVehicle()
 {
 };
 int getmileagefees()
 { double mileagefees ;
 if (kms < 100)
```

```
{
mileagefees = 0.2 * kms;
return mileagefees;
}
else if (400 >= kms >= 100)
{
mileagefees = 0.3 * kms;
return mileagefees;
}
else if (kms > 400)
{
mileagefees = (0.3 * 400) + 0.5 * (kms - 400);
return mileagefees;
}
}
int getkms()
{
return kms;
}
void setkms(int kingsman)
{
kms = kingsman;
}
};
```

```
class Car : public FuelVehicle
{
 int seats;

public:
 Car()
 {
 };
 rentedVehicle civic;
 Car(int r,int l , double c){
 seats=r;
 setkms(l);
 civic.setCost(c);
 }
 virtual int getCost()
 {
 int a = civic.getCost();
 return getmileagefees() + (seats * a);
 }
 int getseats()
 {
 return seats;
 }
 void setseats(int s)
```

```
 {
 seats = s;
 }
};
class Truck : public FuelVehicle
{
 int capacity;

public:
 Truck()
 {
 };
 rentedVehicle civic;
 FuelVehicle fv;
 Truck(int c,int k , double b){
 capacity=c;
 fv.setkms(k);
 civic.setCost(b);
 }
 virtual int getCost()
 {
 int a = civic.getCost();
 return getmileagefees() + (capacity * a);
 };
```

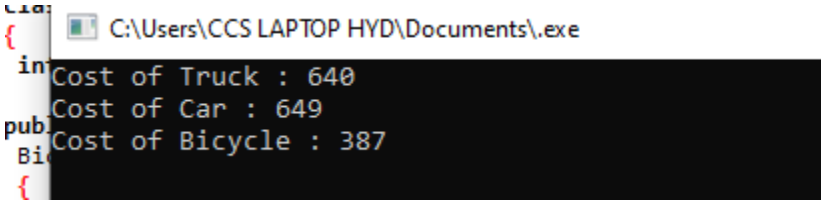
```
 int getcapacity()
 {
 return capacity;
 }
 void setcapacity(int c)
 {
 capacity = c;
 }
};
class Bicycle : public rentedVehicle
{
 int nDays;

public:
 Bicycle()
 {
 nDays = 0;
 };
 rentedVehicle civic;
 Bicycle(int n, double b){
 nDays=n;
 civic.setCost(b);
}
 virtual int getCost()
```

```
 {
 int a = civic.getCost();
 return nDays * a;
 };
 int getnDays()
 {
 return nDays;
 }
 void setnDays(int n)
 {
 nDays = n;
 }
};
int main()
{
Truck T(8,350,80);
 Car C(12,6,54);
 Bicycle B(43,9);

 cout <<"Cost of Truck : "<<T.getCost()<<endl;
 cout <<"Cost of Car : "<<C.getCost()<<endl;
 cout <<"Cost of Bicycle : "<<B.getCost()<<endl;
 return 0;
}
```

OUTPUT:


C:\Users\CCS LAPTOP HYD\Documents\.exe
Cost of Truck : 640
Cost of Car : 649
Cost of Bicycle : 387