Assignment no 5

Q1

```
using namespace std;
#include<iostream>
#include<string.h>

struct Employee{
        int id;
        char name[20];
   double salary;
   virtual void display(){
                cout<<"\n ID = "<<this->id;
        cout<<"\n Name = "<<this->name;
        cout<<"\n salary = "<<this->salary;

   }
   void setid(int a){
                this->id=a;
        }
        void setname(char* ass){
                strcpy(this->name,ass);
```

```
        }
        void setsalary(double a){

       this->salary=a;
        }
        int getid(){
                return this->id;
        }
          char* getname(){
                return this->name;
           }
                double getsalary(){
                return this->salary;
        }
    virtual double cal_salary(){
                return this->salary;
        }

        Employee(){
        //      cout<<"Employee Defoult counstructor call
"<<endl;
           this->id=0;
```

```
        strcpy(this->name,"not given");
        this->salary=0;
    }
    Employee(int id,char*name,double salary){
    //    cout<<"Employee parametrize counstructer call"<<endl;
        this->id=id;
        strcpy(this->name,name);
        this->salary=salary;
    }
};// Employ End here
struct Salsemanager:public Employee{
    // this is frist step is a relationship
    double incentive;
    int target;
    void setincentive(double a){
            this->incentive=a;
    }
            void settarget(int a){
            this->target=a;
    }
    double getincentive(){
```

```
        return this->incentive;
    }
    int gettarget(){
        return this->target;
    }
    Salsemanager():Employee(){



        this->incentive=0.00;
        this->target=00;
    }
  Salsemanager(int id,char*name,double salary,double
incentive ,int target):Employee(id,name,salary){


      this->incentive=incentive;
      this->target=target;
    }
    void display(){
     Employee::display();


   cout<<"\n incentiv = "<<this->incentive;
   cout<<"\n target = "<<this->target;
```

```
  }
      double cal_salary(){
            return this->salary+this->incentive;
      }
};// salse manager end here
struct Hr:public Employee{
      double  commission;
 void setcommission(double a){
            this->commission=a;
      }
 double getcommission(){
            return this->commission;
      }
      Hr():Employee(){

            this->commission=0;
      }
      Hr(int id,char* name,double salary,double
commission):Employee(id,name,salary){
            this->commission=commission;
      }
      void display(){
```

```
        Employee::display();
    cout<<"\n commission = "<<this->commission<<endl;
    }
    double cal_salary(){
        return this->salary+this->commission;
    }

};// Hr ends here
 struct Admin:public Employee{
    double allowance;
    void setallowance(double a){
        this->allowance=a;
     }
     double getallowance(){
        return this->allowance;
     }
     Admin():Employee(){

        this->allowance=0;
     }
     Admin(int id,char* name,double salary,double
allowance):Employee(id,name,salary){
```

```
        this->allowance=allowance;
    }
    void display(){
        Employee::display();
    cout<<"\n allowance = "<<this->allowance<<endl;
    }
    double cal_salary(){
        return this->salary+this->allowance;
    }
}; //Admin end here
int main(){
Employee*ep;
Employee e1(11,"Ashutosh",200000);
ep=&e1;
//ep->display();

Salsemanager S(12,"virat",52635,45236,58);
ep=&S;
 ep->display();
 cout<<"total salary of Salsemanager is = "<<ep-
>cal_salary()<<endl;
```

```
Admin A(13,"sachin",42563,52365);
ep=&A;
ep->display();
cout<<"total salary of admin is = "<<ep->cal_salary()<<endl;

 Hr H(14,"virat",1256,5236);
 ep=&H;
 ep->display();
 cout<<"total salary of Hr is = "<<ep->cal_salary()<<endl;

}
```

Q2

```
using namespace std;
#include<iostream>
struct Shape{
        float area;
        char colour[20];
        virtual void display(){
                cout<<"Area :"<<this->area<<endl;
      cout<<"Colour : "<<this->colour<<endl;
```

```
}
virtual float cal_area(){
        return this->area;
}
void setarea(float a){
        this->area=a;
}
void setcolour(char* str){
        strcpy(this->colour,str);
}
float getarea(){
        return this->area;
}
char* getcolour(){
        return this->colour;
}
Shape(){
        this->area=0.0;
        strcpy(this->colour,"not given");
}
Shape(float a,char*colour){
```

```
            this->area=a;
            strcpy(this->colour,colour);
        }
};
struct Trangle:public Shape{
        float hight;
        float brith;
        void display(){
            Shape::display();
            cout<<"Hight : "<<this->hight<<endl;
            cout<<"brith : "<<this->brith<<endl;

        }
           void sethight(float a){
            this->hight=a;
           }

             void setbrith(float b){
            this->brith=b;
            }
            float gethight(){
                    return this->hight;
```

```
        }
        float getbrith(){
            return this->brith;
        }
        Trangle():Shape()
        {
            this->hight=00;
            this->brith=00;
        }
        Trangle(float area,char*colour,float hight,float brith
):Shape(area,colour)
        {
            this->hight=hight;
            this->brith=brith;
        }
     float cal_area(){
            return this->area=0.5*(this->hight)*(this->brith);
    }

};
struct Rectangle:public Shape{
    float length;
```

```
float width;
void display(){
        Shape::display();
        cout<<"Length : "<<this->length<<endl;
        cout<<"Width : "<<this->width<<endl;

}
    void setlength(float a){
        this->length=a;
        }
         void setwidth(float b){
        this->width=b;
        }
        float getlength(){
                return this->length;
        }
        float getwidth(){
                return this->width;
        }
         Rectangle():Shape(){
                this->length=00;
                this->width=00;
```

```
        }
        Rectangle(float area,char*colour,float lenght,float
width):Shape(area,colour){
            this->length=lenght;
            this->width=width;
        }
    float cal_area(){
        this->area=(this->length)*(this->width);
    }
};
struct Circle:public Shape{
    float radius;
    void display(){
      Shape::display();
        cout<<"radius : "<<this->radius<<endl;
    }
    void setradius(float a){
        this->radius=a;
    }
       float getradius(){
        return this->radius;
    }
```

```
        Circle():Shape(){
                this->radius=0;
        }
                Circle(float area,char*colour,float
radius):Shape(area,colour){
                this->radius=radius;
        }
         float cal_area(){
                return this->area=(3.14*3.14)*this->radius;
        }

};
 int main(){
    Shape *ptr;
        Trangle T1(8.5f,"black",4.5f,6.3f);
  ptr=&T1;
  ptr->display();
  cout<<"total area of trangle is = "<<ptr->cal_area()<<endl;


    Rectangle R1(8.6f,"white",78.5f,5.3f);
    ptr=&R1;
```

```
  ptr->display();
  cout<<"total area of Rectangle is = "<<ptr->cal_area()<<endl;



    Circle C1(7.5f,"greay",4.5);
    ptr=&C1;
  ptr->display();
  cout<<"total area of Circle is = "<<ptr->cal_area()<<endl;


}
```

Q3

```
using namespace std;
#include<iostream>
#include<string.h>
struct Vehicle{
        char modelname[30];
        double price;
        char colour[20];
        int yearofmanu;
    virtual void display(){
        cout<<"Model name = "<<this->modelname<<endl;
        cout<<"price = "<<this->price<<endl;
```

```
        cout<<"colour = "<<this->colour<<endl;
        cout<<"Year of manufacturing = "<<this-
>yearofmanu<<endl;
        }
        void setmodelname(char*ptr){
                strcpy(this->modelname,ptr);
        }
        void setprice(double a){
                this->price=a;
        }
        void setcolour(char*ptr){
                strcpy(this->colour,ptr);
        }
        void setyearofmanu(int a){
                this->yearofmanu;
        }
        char* getmodelname(){
                return this->modelname;
        }
        double getprice(){
                return this->price;
                }
```

```
        char* getcolour(){
                return this->colour;
        }
        int getyearofmanu(){
                return this->yearofmanu;
        }
        Vehicle(){
                strcpy(this->modelname,"not given");
                this->price=0.0;
                strcpy(this->colour,"not give");
                this->yearofmanu=0000;
        }
        Vehicle(char*str,double a,char*ptr,int b){
                strcpy(this->modelname,str);
                this->price=a;
                strcpy(this->colour,ptr);
                this->yearofmanu=b;
        }
        virtual void brake() {
       cout<<"Applying brakes."<<endl;
    }
};//vehicle end here
```

```
struct Car:public Vehicle{
        int no_of_airbag;
        int no_of_ac;
        void display(){
         Vehicle::display();
        cout<<"no of airbag = "<<this->no_of_airbag<<endl;
        cout<<"no of ac = "<<this->no_of_ac<<endl;
                }
                void setno_of_airbag(int a){
                        this->no_of_airbag=a;
                }
                void setno_of_ac(int b){
                        this->no_of_ac;
                }
                int getno_of_airbag(){
                        return this->no_of_airbag;
                }
                int getno_of_ac(){
                        return this->no_of_ac;
                }
                Car():Vehicle(){
                        this->no_of_airbag=0;
```

```
                this->no_of_ac=0;
        }
                Car(char*modelname,double
price,char*colour,int year,int airbag,int
ac):Vehicle(modelname,price,colour,year){
                this->no_of_airbag=airbag;
                this->no_of_ac=ac;
        }
        void brake(){
      cout<<"Activating drum brake."<<endl;
   }

};
 struct Bick:public Vehicle{
       int no_of_stand;
       void display(){
       Vehicle::display();
       cout<<"no of stand = "<<this->no_of_stand<<endl;
       }
       void setno_of_stand(int a){
              this->no_of_stand=a;
       }
```

```
        int getno_of_stand(){
                return this->no_of_stand;
        }
        Bick():Vehicle(){
                this->no_of_stand=0;
        }
                Bick(char*modelname,double price,char*colour,int
year,int stand):Vehicle(modelname,price,colour,year){
                this->no_of_stand=stand;
        }
        void brake(){
       cout<<"Activating disc brake."<<endl;
    }

 };// bick end here
 struct Bus:public Vehicle{
        char type_of_bus[20];
 void display(){
        Vehicle::display();
        cout<<"type of bus(city bus/school bus/luxury coach =
"<<this->type_of_bus<<endl;
        }
```

```
        void settype_of_bus(char*str){
                strcpy(this->type_of_bus,str);
        }
        char* gettype_of_bus(){
                return this->type_of_bus;
        }
        Bus():Vehicle(){
                strcpy(this->type_of_bus,"not given");
        }
        Bus(char*modelname,double price,char*colour,int
year,char*bustype):Vehicle(modelname,price,colour,year){
                strcpy(this->type_of_bus,bustype);
        }
        void brake(){
       cout<<"Activating air brake."<<endl;
   }


 };
int main(){
        Vehicle *ptr;
        Car v2("suv",85212.5,"red",2002,4,2);
        ptr=&v2;
```

```
    ptr->display();
    ptr->brake();



    Bick B1("tvs Appache",150000,"black",2024,2);
    ptr=&B1;
    ptr->display();
    ptr->brake();



    Bus S1("Tata",852365,"white",2019,"Schoole bus");
    ptr=&S1;
    ptr->display();
    ptr->brake();

}
```

Q4

```
using namespace std;
#include<iostream>
#include<string.h>
struct HomeAppliance{
```

```
        char company_nm[20];
        char colour[20];
        double weight;
        double price;
        virtual  void display(){
                cout<<"Company Name = "<<this-
>company_nm<<endl;
                cout<<"Colour = "<<this->colour<<endl;
                cout<<"Weight = "<<this->weight<<endl;
                cout<<"Price = "<<this->price<<endl;
        }
        void setcompany_name(char*str){
                strcpy(this->company_nm,str);
        }
        void setcolour(char*str){
                strcpy(this->colour,str);
        }
        void setweight(double a){
                this->weight=a;
        }
        void setprice(double a){
                this->price=a;
```

```
}
char* getcompany_name(){
        return this->company_nm;
}
char* getcolour(){
        return this->colour;
}
  double getweight(){
        return this->weight;
  }
  double getprice(){
        return this->price;
  }
  HomeAppliance(){
        strcpy(this->company_nm,"not given");
        strcpy(this->colour,"not given");
        this->weight=00;
        this->price=00;
  }
  HomeAppliance(char*str,char*str2,double a,double b){

        strcpy(this->company_nm,str);
```

```
            strcpy(this->colour,str2);
            this->weight=a;
            this->price=b;
        }
        virtual int warrantyPeriod() {
       return 0;
   }


};// HomeAppliance end here
struct WashingMachine:public HomeAppliance{
        int water_con;
        int capacity;
        void setwater_con(int a){
                this->water_con=a;
        }
        void setcapacity(int a){
                this->capacity=a;
        }
        int getwater_con(){
                return this->water_con;
        }
```

```
int getcapacity(){
    return this->capacity;
}
WashingMachine(){
    this->water_con=0;
    this->capacity=0;
}
WashingMachine(char*str,char*str2,double a,double b,int c,int d):HomeAppliance(str,str2,a,b){
    this->water_con=c;
    this->capacity=d;
}
void display(){
    HomeAppliance::display();
    cout<<"Water consumption = "<<this->water_con<<endl;
    cout<<"Capacity = "<<this->capacity<<endl;
}
 int warrantyPeriod() {
if(this->getprice()>=10000){
 return 5;
    }
```

```
            else if(this->getprice()>=70000){
                return 3;
            }
            else{
                return 1;
            }
    }
};// WashingMachine end here
struct Refrigerator:public HomeAppliance{
    float energyrating;
    int no_ofdoors;
     void display(){
    HomeAppliance::display();
    cout<<"Energy Rating = "<<this->energyrating<<endl;
    cout<<"No of Doors = "<<this->no_ofdoors<<endl;
     }
    void setenergyrating(float a){
            this->energyrating=a;
     }
     void setno_ofdoors(int a){
            this->no_ofdoors=a;
     }
```

```
 float getenergyrating(){
        return this->energyrating;
      }
       int setno_ofdoors(){
        return this->no_ofdoors;
      }
      Refrigerator(){
        this->energyrating=0;
        this->no_ofdoors=0;
      }
      Refrigerator(char*str,char*str2,double a,double b,float
c,int d):HomeAppliance(str,str2,a,b){
        this->energyrating=c;
        this->no_ofdoors=d;
      }
       int warrantyPeriod() {
     if(this->getprice()>=15000){
      return 4;
        }
        else if(this->getprice()>=90000){
            return 2;
        }
```

```
            else{
                return 1;
            }
}

};//Refrigerator end here
struct Microwave:public HomeAppliance{
    int cookingpower;
void setcookingpower(int a){
            this->cookingpower=a;
    }
    int getcookingpower(){
            return this->cookingpower;
    }
    Microwave(){
            this->cookingpower=0;
    }
    Microwave(char*str,char*str2,double a,double b,int
c):HomeAppliance(str,str2,a,b){
            this->cookingpower=c;
    }
    void display(){
```

```
            HomeAppliance::display();
            cout<<"Cooking power = "<<this-
>cookingpower<<endl;
        }
         int warrantyPeriod() {
        if(this->getprice()>=80000){
         return 3;
            }
            else if(this->getprice()>=50000){
                return 2;
            }
            else{
                return 1;
            }
}

};


int main(){
 HomeAppliance *ptr;
 WashingMachine W1("qulitybuild","Black",263,8523,5,8);
```

```
 ptr=&W1;
ptr->display();
cout<<"Warranty period is = "<<ptr-
>warrantyPeriod()<<"Year"<<endl;



Refrigerator R1("samsung","Red",785,17852,4.3f,2);
ptr=&R1;
ptr->display();
cout<<"Warranty period is = "<<ptr-
>warrantyPeriod()<<"Year"<<endl;



Microwave M1("freshfood","white",125,5896,12);
ptr=&M1;
 ptr->display();
cout<<"Warranty period is = " <<ptr-
>warrantyPeriod()<<"Year"<<endl;



}
```

Second Example

```
using namespace std;
#include<iostream>
struct Company{
        char name[30];
        char manufacturing[40];
        int yearof_esta;
        int no_ofemp;
        double turnover;
  virtual void display(){
        cout<<"Company name = "<<this->name<<endl;
        cout<<"Manufacturing = "<<this->manufacturing<<endl;
        cout<<"Year of Established = "<<this-
>yearof_esta<<endl;
        cout<<"Employee Count = "<<this->no_ofemp<<endl;
        cout<<"Turnover = "<<this->turnover<<endl;
   }
  void setname(char*str){
        strcpy(this->name,str);
   }
   void set_manufacturing(char*str){
        strcpy(this->manufacturing,str);
```

```
}
void setyearof_esta(int a){
    this->yearof_esta=a;
}
void setno_ofemp(int a){
    this->no_ofemp=a;
}
void setturnover(int a){
    this->turnover=a;
}
char* getname(){
    return this->name;
}
char* get_manufacturing(){
    return this->manufacturing;
}
int getyearof_esta(){
    return this->yearof_esta;
}
int getno_ofemp(){
    return this->no_ofemp;
}
```

```
  int getturnover(){
      return this->turnover;
  }
  Company(){
      strcpy(this->name,"not give");
      strcpy(this->manufacturing,"not given");
      this->yearof_esta=00;
      this->no_ofemp=00;
      this->turnover=00;
  }
   Company(char *name,char*manu,int year,int emp,int
turn){
      strcpy(this->name,name);
      strcpy(this->manufacturing,manu);
      this->yearof_esta=year;
      this->no_ofemp=emp;
      this->turnover=turn;
  }
      virtual double calculateRevenue(){
     return this-> getturnover()*0.2;
   }
};
```

```
struct It_Company:public Company{
	char datacenter[40];
	char techstack[50];
	int ongoing_projects;
	void display(){
		Company::display();
		cout<<"location of datacenters = "<<this->datacenter<<endl;
		cout<<"List of technologies used = "<<this->techstack<<endl;
		cout<<"No of current projects = "<<this->ongoing_projects<<endl;
	}
	void setdatacenter(char*str){
		strcpy(this->datacenter,str);
	}
	void settechstack(char*str){
			strcpy(this->techstack,str);
	}
	void setproject(int a){
		this->ongoing_projects=a;
	}
	char* getdatacenter(){
```

```
        return this->datacenter;
    }
    char* gettechstack(){
            return this->techstack;
    }
    int getproject(){
        return this->ongoing_projects;
    }
    It_Company():Company(){
        strcpy(this->datacenter,"not given");
        strcpy(this->techstack,"not given");
        this->ongoing_projects=0;
    }
    It_Company(char *name,char*manu,int year,int emp,int
turn,char*center,char*tech,int
project):Company(name,manu,year,emp,turn){
        strcpy(this->datacenter,center);
        strcpy(this->techstack,tech);
        this->ongoing_projects=project;
    }
 double calculateRevenue() {
    return this->getturnover()*2.2;
```

```
    }
};
struct ClothManufacturingCompany:public Company{
        char fabricTypes[30];
        int textileWaste;
        void display(){
                Company::display();
                cout<<"Fabric types = "<<this->fabricTypes<<endl;
                cout<<"Textile Waste = "<<this-
>textileWaste<<endl;
        }
        void setfabric(char *str){
                strcpy(this->fabricTypes,str);
 }
 void setwaste(int a){
        this->textileWaste=a;
 }
 char* getfabric(){
                return this->fabricTypes;
 }
 int setwaste(){
        return this->textileWaste;
```

```
 }
 ClothManufacturingCompany():Company()
 {
        strcpy(this->fabricTypes,"not given");
        this->textileWaste=0;
 }
 ClothManufacturingCompany(char *name,char*manu,int
year,int emp,int turn,char*type,int
waste):Company(name,manu,year,emp,turn){
        strcpy(this->fabricTypes,type);
        this->textileWaste=waste;
 }
double calculateRevenue() {
       return this->getturnover()*1.2;
   }
};
int main(){
        Company *ptr;

        It_Company
C("Techbull","Software",2015,45,845964,"pune","Java,spring,
mysql",13);
   ptr=&C;
```

```
        ptr->display();
        cout<<"Revenue of IT Company = "<<ptr-
>calculateRevenue()<<endl;


        ClothManufacturingCompany S("Rowdy","Man
Cloths",2009,78,784587,"Cotton",100);
ptr=&S;
ptr->display();
cout<<"Revenue of IT Company = "<<ptr-
>calculateRevenue()<<endl;


}
```