Name : Muhammad usama

Roll : 17f\_8915

Section : B

Lab : 9

Task 1:

#include<iostream>

#include<string>

using namespace std;

class employee

{

protected:

string name;

string id;

public:

virtual void pay()=0;

void setname(string n)

{

name=n;

}

void setid(string i)

{

id=i;

}

};

class piece\_worker: public employee

{

protected:

int wage;

int nopieces;

public:

piece\_worker(int w, int np)

{

wage=w;

nopieces=np;

}

void setwage(int w)

{

wage=w;

}

void setnopiece(int p)

{

nopieces=p;

}

int getwage()

{

return wage;

}

int getnopieces()

{

return nopieces;

}

void pay()

{

cout<<"pay : "<<wage\*nopieces;

}

};

class hours\_worker: public employee

{

private:

int wage;

int nohours;

int pay1;

public:

hours\_worker(int w,int nh)

{

wage=w;

nohours=nh;

}

void pay()

{

if(nohours<=40)

{

cout<<"pay : "<<wage\*nohours;

}

else

{

pay1=wage\*40;

pay1=pay1+(1.5)\*wage\*(nohours-40);

cout<<"pay : "<<pay1;

}

}

};

int main()

{

int w;

int np;

int nh;

string name;

string id;

employee \*ptr[2];

cout<<"enetr name of person : ";

cin>>name;

cout<<"eneter id of person : ";

cin>>id;

cout<<"eneter wage of piece worker : ";

cin>>w;

cout<<"enetr no of pieces : ";

cin>>np;

ptr[0]=new piece\_worker(w,np);

//ptr[0]->pay();

ptr[0]->setid(id);

ptr[0]->setname(name);

ptr[0]->pay();

cout<<endl;

cout<<"enetr name of person : ";

cin>>name;

cout<<"eneter id of person : ";

cin>>id;

cout<<"eneter wage of hourly worker : ";

cin>>w;

cout<<"enetr no of hours : ";

cin>>nh;

ptr[1]=new hours\_worker(w,nh);

ptr[1]->setid(id);

ptr[1]->setname(name);

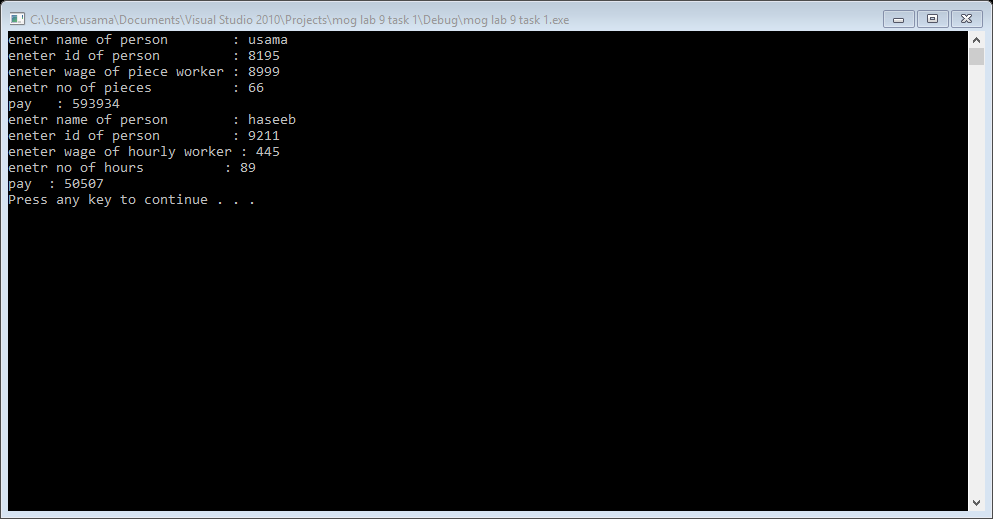
ptr[1]->pay();

cout<<endl;

system("pause");

return 0;}

Result:



Task 2:

#include<iostream>

#include<tchar.h>

using namespace std;

class polygon

{

protected:

float length;

float width;

float a;

float p;

public:

virtual float area()=0;

virtual float perimeters()=0;

virtual void display()=0;

};

class square : public polygon

{

public:

square(float l ,float w)

{

length=l;

width=w;

}

square()

{ }

float area()

{

a=length\*width;

return a;

}

float perimeters()

{

p=4\*length;

return p;

}

void display()

{

cout<<"area of square : "<<a<<endl;

cout<<"paramiters of square : "<<p<<endl;

}

};

class rectangle:public polygon

{

public:

rectangle(float l ,float w)

{

length=l;

width=w;

}

rectangle()

{ }

float area()

{

a=2\*(length\*width);

return a;

}

float perimeters()

{

p=4\*length;

return p;

}

void display()

{

cout<<"area of rectangle : "<<a<<endl;

cout<<"parameters of rectangle : "<<p<<endl;

}

};

class triangle :public polygon

{

public:

triangle(float l ,float w)

{

length=l;

width=w;

}

triangle()

{ }

float area()

{

a=1/2\*(length\*width);

return a;

}

float perimeters()

{

p=2\*length+width;

return p;

}

void display()

{

cout<<"area of triangle : "<<a<<endl;

cout<<"parameters of rectangle : "<<p<<endl;

}

};

int main()

{

int ch;

float l;

float w;

do

{

cout<<"1 :calculate area of aquare "<<endl;

cout<<"2 :caculate area of rectangle"<<endl;

cout<<"3 :caculate area of triangle"<<endl;

cin>>ch;

polygon \*ptr[3];

if(ch==1)

{

cout<<"eneter length of aquare : ";

cin>>l;

cout<<"enetr width of square : ";

cin>>w;

ptr[0]=new square(l,w);

ptr[0]->area();

ptr[0]->perimeters();

ptr[0]->display();

}

else if(ch==2)

{

cout<<"eneter length of rectangle : ";

cin>>l;

cout<<"enetr width of rectangle : ";

cin>>w;

ptr[1]=new rectangle(l,w);

ptr[1]->area();

ptr[1]->perimeters();

ptr[1]->display();

}

else if(ch==3)

{

cout<<"eneter length of triangle : ";

cin>>l;

cout<<"enetr width of triangl : ";

cin>>w;

ptr[2]=new triangle(l,w);

ptr[2]->area();

ptr[2]->perimeters();

ptr[2]->display();

}

else

{

system("cls");

ch=0;

cout<<"invalid choice"<<endl;

cout<<"please enter again ";

}

cout<<"if you calculate any other press 0 :";

cin>>ch;

system("cls");

}

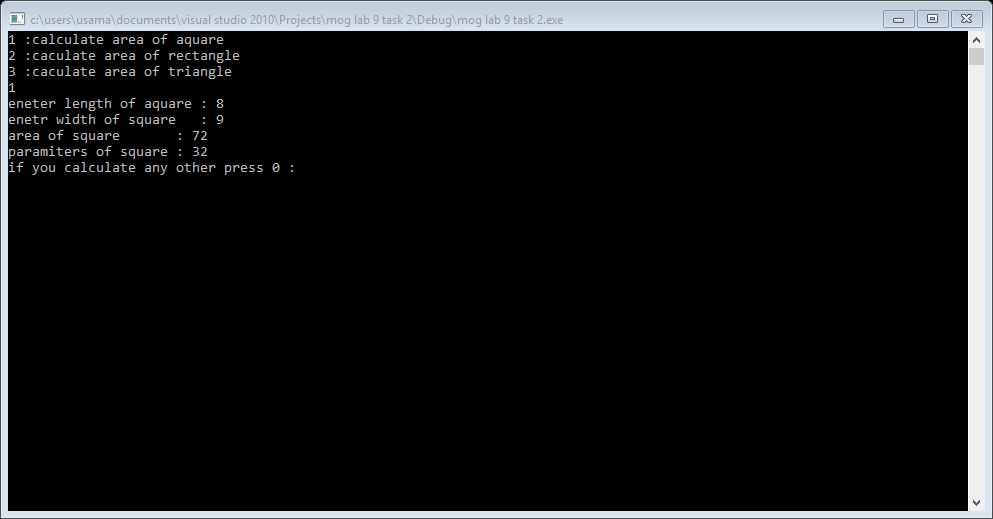
while(ch==0);

system("pause");

return 0;

}

Result:



Task 3:

#include<iostream>

#include<string>

using namespace std;

class employee

{

private:

string name;

int rank;

int bpay;

int md;

int hrent;

int gpay;

void display()

{

cout<<"name : "<<this->name<<endl;

cout<<"Rank : "<<this->rank<<endl;

cout<<"basic pay : "<<this->bpay<<endl;

cout<<"medical allowance : "<<this->md<<endl;

cout<<"house rent : "<<this->hrent<<endl;

cout<<"grass pay : "<<this->gpay<<endl;

}

public:

employee():name(""),rank(0),bpay(0),md(0),hrent(0),gpay(0)

{

cout<<"constructor"<<endl;

}

friend void read\_record(employee);

friend void gross\_pay(employee);

friend void a\_increment(employee);

~employee()

{

cout<<"destructor"<<endl;

}

};

void read\_record(employee e)

{

cout<<"enter name of employee : ";

cin>>e.name;

cout<<"enetr rank of employee : ";

cin>>e.rank;

cout<<"enetr basic pay of employee : ";

cin>>e.bpay;

e.md=0.60\*e.bpay;

e.hrent=(28.9/100)\*e.bpay;

e.gpay=e.bpay+e.md;

e.display();

}

/\*void gross\_pay(employee e)

{

e.gpay=e.bpay+e.md;

}

void a\_increment(employee e)

{

e.bpay=e.bpay+20/100\*(e.bpay);

e.display();

}\*/

int main()

{

employee e;

/\*employee g;

employee a;

read\_record(r);

gross\_pay(g);

a\_increment(a);\*/

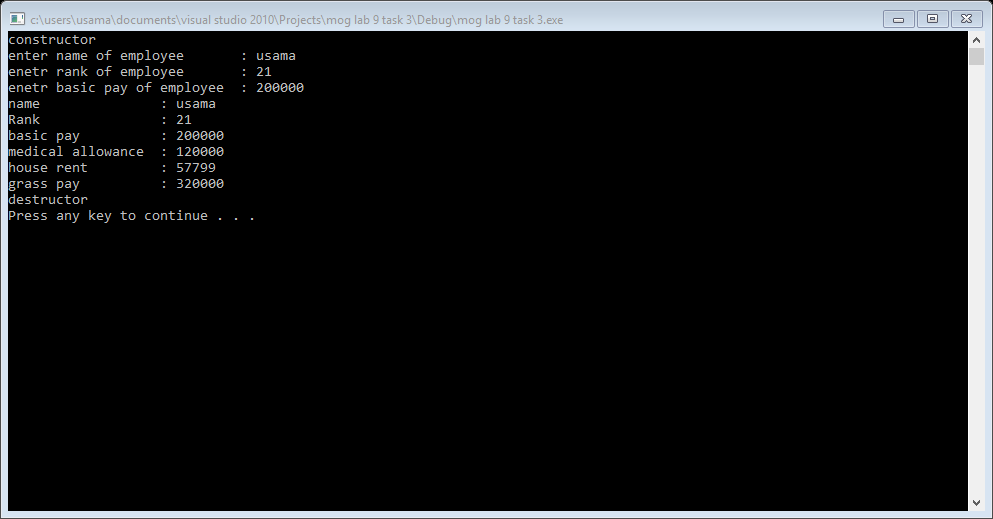
read\_record(e);

system("pause");

return 0;

}

Result:



Task 4:

#include<iostream>

using namespace std;

class base

{

private:

int base1;

int base2;

int base3;

public:

base(int b1,int b2,int b3)

{

base1=b1;

base2=b3;

base3=b3;

}

base():base1(0),base2(0),base3(0)

{ }

void display()

{

cout<<"base1 is : "<<base1<<endl;

cout<<"base2 is : "<<base2<<endl;

cout<<"base3 is : "<<base3<<endl;

}

friend class friend\_class;

};

class derive

{

private:

int derive1;

int derive2;

public:

derive(int d1, int d2)

{

derive1=d1;

derive2=d2;

}

derive():derive1(1),derive2(1)

{ }

friend class friend1\_class;

};

class friend\_class

{

private:

int f1;

int f2;

friend class friend2\_class;

public:

void display(base b)

{

cout<<"base 1 is :"<<b.base1<<endl;

cout<<"base 2 is :"<<b.base2<<endl;

cout<<"base 3 is :"<<b.base3<<endl;

}

friend\_class()

{

f1=0;

f2=0;

}

};

class friend1\_class

{

public:

void display(derive d)

{

cout<<"derive1 is :"<<d.derive1<<endl;

cout<<"derive2 is :"<<d.derive2<<endl;

/\*cout<<"base 1 is :"<<d.base1<<endl; //not accees able

cout<<"base 2 is :"<<d.base2<<endl;\*/

}

};

class friend2\_class

{

public:

void display(friend\_class f)

{

cout<<"f1 is : "<<f.f1<<endl;

cout<<"f2 is : "<<f.f2<<endl;

}

};

int main()

{

base b(1,2,3);

derive d(4,5);

friend\_class f;

f.display(b);

friend1\_class f1;

f1.display(d);

friend2\_class f2;

f2.display(f);

system("pause");

return 0;

}

Result:

