Problem Statement: You have the create an inheritance among **Father**-->**Son** -->**GrandSon** class. The **father** class has the following data members

```
 class Father{
private:
int money;
protected:
int gold;
public:
};
```

Now write the **Son** and **GrandSon** classes with **private**/**protected**/**public** access modifier and do the following:

i) Try to access **money**, **gold** and **land** from Son class ii) Try to access **money**, **gold** and **land** from GrandSon class

iii) Find the values of money, gold and land when different access modifer is used in the following table .

Class		In Son class			In GrandSon class		
Son	**GrandSon**	**money**	**gold**	**land**	**money**	**gold**	**land**
public	public	?	?	?	?	?	?
protected	public	?	?	?	?	?	?
private	public	?	?	?	?	?	?
public	protected	?	?	?	?	?	?
protected	protected	?	?	?	?	?	?
private	protected	?	?	?	?	?	?
public	private	?	?	?	?	?	?
protected	private	?	?	?	?	?	?
private	private	?	?	?	?	?	?

Solution:

Theory:

Object oriented programming has a characteristic named inheritance.Inheritance means a class can inherit data members of a class by certain syntax and access specifiers. There are three types of access specifiers.

1. Public
2. Protected
3. Private

Publicly inherited child class inherits base classes all data members as they are in the base class means public members as public and protected members as protected.

Inherited as protected makes all the inherited data members of base class protected in child class. And privately inherited child class inherits all the data members of base class as private members of own.
Private members of a class can never be inherited.

<u>Son and grandson class will inherit:</u>

Class		**In Son class**			**In GrandSon class**		
Son	**GrandSon**	**money**	**gold**	**land**	**money**	**gold**	**land**
public	public	×	√	√	×	√	√
protected	public	×	√	√	×	√	√
private	public	×	√	√	×	×	×
public	protected	×	√	√	×	√	√
protected	protected	×	√	√	×	√	√
private	protected	×	√	√	×	×	×
public	private	×	√	√	×	√	√
protected	private	×	√	√	×	√	√
private	private	×	√	√	×	×	×

<u>Problem 1:</u>

<u>Inheritance Son(Publicly) & Grandson(Publicly):</u>

```
#include<iostream>
using namespace std;
class father
{
   int money; protected:
```

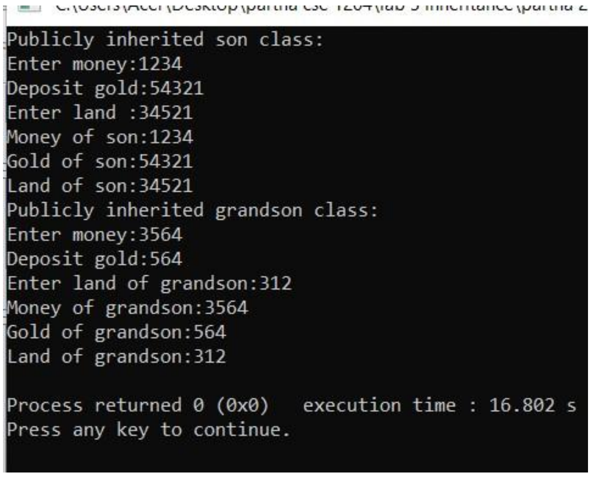
```
  int gold; public:     int
land;     void
set_money(){     cout
<<"Enter money:";
  cin>>money;
  }
  int get_money(){
  return money;
  }
};
class son : public   father
{
public:
  int x;
  void set_gold_son()
  {
    cout<<"Deposit gold:";
    cin >>x;
    gold=x;
  }
  int get_gold_son()
  {
    return gold;
  }
};
class grandson: public son
{
public:
  int x;
};
int main()
{     son ob;
  cout<<"Publicly inherited son class:\n";
ob.set_money();     ob.set_gold_son();
cout<<"Enter land :";
  cin>>ob.land;
  cout<<"Money of son:"<<ob.get_money()<<endl; cout<<"Gold of son:"<<ob.get_gold_son()<<endl;
cout<<"Land of son:"<<ob.land<<endl; cout<<"Publicly inherited grandson class:\n"; grandson
ob_grandson;
   ob_grandson.set_money();
ob_grandson.set_gold_son();
cout<<"Enter land of grandson:";
cin>>ob_grandson.land;
  cout<<"Money of
grandson:"<<ob_grandson.get_money()<<endl;
cout<<"Gold of
grandson:"<<ob_grandson.get_gold_son()<<endl;
```

```
cout<<"Land of grandson:"<<ob_grandson.land<<endl;
return 0;
}
```

Discussion :

As money is private in the father class it needed a public function to give access to the child and grandchild class. Other members were inherited as their access specifier cause both child and grandson were inherited publicly .But gold needed function in child class to access by object of child and grandchild class.

Output:



```
Publicly inherited son class:
Enter money:1234
Deposit gold:54321
Enter land :34521
Money of son:1234
Gold of son:54321
Land of son:34521
Publicly inherited grandson class:
Enter money:3564
Deposit gold:564
Enter land of grandson:312
Money of grandson:3564
Gold of grandson:564
Land of grandson:312

Process returned 0 (0x0)   execution time : 16.802 s
Press any key to continue.
```

Problem 2:

Inheritance Son(As Protected) & Grandson(Publicly) :

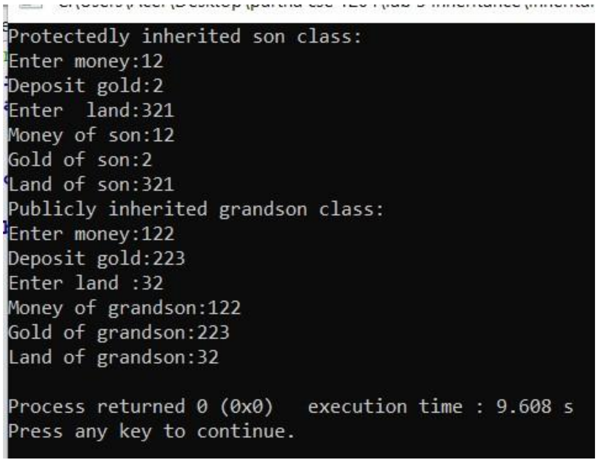
```
#include<iostream>
using namespace std;
class father
{
   int money;
protected:
int gold;
public:
   int land;
   int set_money()
   {
      cout <<"Enter money:";
      cin>>money;
   }
   int get_money()
   {
      return money;
   }
};
class son : protected   father
```

```
{
public:     int
get_money_son()
   {
      get_money();
   }
   int set_money_son()
   {
      set_money();
   }
   int x;
   void set_gold_son()
   {
      cout<<"Deposit  gold:";
cin >>x;        gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
   void set_land_son()

{        cin >>
x;
      land=x;
   }
   int get_land_son()
   {
      return land;
   }
};
class grandson: public son
{
public:
};
int main()
{     son ob;
   cout<<"Protectedly inherited son class:\n";
ob.set_money_son();         ob.set_gold_son();
cout<<"Enter  land:";
    ob.set_land_son(); cout<<"Money of
    son:"<<ob.get_money_son()<<endl; cout<<"Gold of
    son:"<<ob.get_gold_son()<<endl; cout<<"Land of
    son:"<<ob.get_land_son()<<endl;
   cout<<"Publicly inherited grandson class:\n";     grandson ob2;
ob2.set_money_son();     ob2.set_gold_son();     cout<<"Enter
land :";     ob2.set_land_son();     cout<<"Money of
grandson:"<<ob2.get_money_son()<<endl;     cout<<"Gold of
grandson:"<<ob2.get_gold_son()<<endl;     cout<<"Land of
grandson:"<<ob2.get_land_son()<<endl;     return 0;
}
```

Discussion :

As money is private in the father class it needed a public function to give access to the child and grandchild class. Other members were inherited as protected in child class and grandson class .So all members needed function in child class to access by object of child and grandchild class because grandchild class inherited child classes members using public access specifier.

Output:



```
Protectedly inherited son class:
Enter money:12
Deposit gold:2
Enter  land:321
Money of son:12
Gold of son:2
Land of son:321
Publicly inherited grandson class:
Enter money:122
Deposit gold:223
Enter land :32
Money of grandson:122
Gold of grandson:223
Land of grandson:32

Process returned 0 (0x0)   execution time : 9.608 s
Press any key to continue.
```

Problem 3:

Inheritance Son(Privately) & Grandson(Publicly):

```
#include<iostream>
 using namespace std;
class father
{
   int money;
protected:
int gold;
public:
   int land;
    void set_money()
    {
       cout <<"Enter money:";
       cin>>money;
    }
   int get_money()
   {
       return money;
   }
};
class son: private  father
{
```

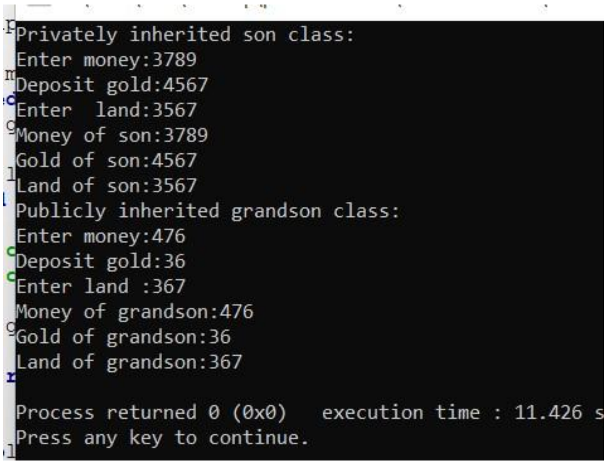
```
public:     int
get_money_son()
   {
      get_money();
   }
   int set_money_son()
   {
      set_money();
   }
   int x;
   void set_gold_son()
   {
      cout<<"Deposit  gold:";
cin >>x;        gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
   void set_land_son()

{        cin >>
x;
      land=x;
   }
   int get_land_son()
   {
      return land;
   }
};
class grandson: public son
{
public:
};
int main()
{     son ob;
    cout<<"Privately inherited son class:\n"; ob.set_money_son();
    ob.set_gold_son();
    cout<<"Enter  land:";
    ob.set_land_son();
    cout<<"Money of son:"<<ob.get_money_son()<<endl;
   cout<<"Gold of son:"<<ob.get_gold_son()<<endl;     cout<<"Land
of son:"<<ob.get_land_son()<<endl;     cout<<"Publicly inherited
grandson class:\n";     grandson ob_grandson;
ob_grandson.set_money_son();     ob_grandson.set_gold_son();
cout<<"Enter land :";     ob_grandson.set_land_son();
cout<<"Money of
grandson:"<<ob_garndson.get_money_son()<<endl;
```

```
cout<<"Gold of grandson:"<<ob_grandson.get_gold_son()<<endl;
cout<<"Land of grandson:"<<ob_grandson.get_land_son()<<endl;
return 0;
}
```

Discussion :

As money is private in the father class it needed a public function to give access to the child and grandchild class. Other members were inherited as private in child class so son class needed public functions to give access to the grandchild class of gold and land. So all members needed function in child class to access by object of child . Grandchild class because grandchild class inherited child classes members using public access specifier.

Output:



```
Privately inherited son class:
Enter money:3789
Deposit gold:4567
Enter  land:3567
Money of son:3789
Gold of son:4567
Land of son:3567
Publicly inherited grandson class:
Enter money:476
Deposit gold:36
Enter land :367
Money of grandson:476
Gold of grandson:36
Land of grandson:367

Process returned 0 (0x0)   execution time : 11.426 s
Press any key to continue.
```

Problem 4:

Inheritance Son(Publicly) & Grandson(As Protected):

```
#include<iostream>
using namespace std;
class father
{
    int money;
protected:     int
  gold;
public:     int land;
  void set_money()
    {
       cout <<"Enter money:";
       cin>>money;
    }
    int get_money()
    {
       return money;
    }
```

```
};
class son : public   father
{
public:
int x;
   int set_gold_son()
   {
      cout<<"Deposit gold:";
cin >>x;
      gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
};
class grandson: protected son
{
public:
int x;
   int set_money_gson()
   {
      set_money();
   }
   int get_money_gson()
   {
      get_money();
   }
   int set_gold_gson()
   {
      set_gold_son();
   }
   int get_gold_gson()
   {
      get_gold_son();
    }
    int set_land_gson()
   {        cin>>l
and;
   }
   int get_land_gsoni()
   {
      return land;
   }
};
int main()
{     son ob;
```

```
   cout<<"Publicly inherited son class:\n";
ob.set_money();     ob.set_gold_son();
cout<<"Enter land :";
   cin>>ob.land;
   cout<<"Money of son:"<<ob.get_money()<<endl;
cout<<"Gold of son:"<<ob.get_gold_son()<<endl;
cout<<"Land of son:"<<ob.land<<endl;
cout<<"Protectedly inherited grandson class:\n";
grandson ob_grandson;
ob_grandson.set_money_gson();
ob_nati.set_gold_gson();     cout<<"Enter land :";
ob_gson.set_land_gson();
   cout<<"Money of grandson:"<<ob_gson.get_money_gson()<<endl;
cout<<"Gold of grandson:"<<ob_gson.get_gold_gson()<<endl;
cout<<"Land of grandson:"<<ob_gson.get_land_gson()<<endl;
return 0;
}
```

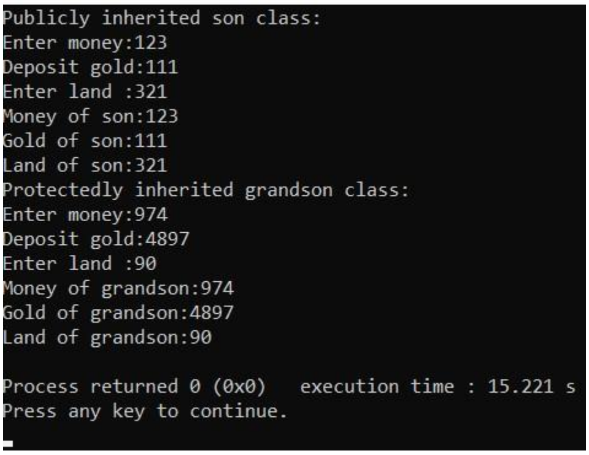
Discussion :

Father: Need function to give access of money.

Son :   Need function to access gold.

Grandson: Need function to access function of gold of son.Need function of land to access land . # # #Need function to access money function of father.

Output:



```
Publicly inherited son class:
Enter money:123
Deposit gold:111
Enter land :321
Money of son:123
Gold of son:111
Land of son:321
Protectedly inherited grandson class:
Enter money:974
Deposit gold:4897
Enter land :90
Money of grandson:974
Gold of grandson:4897
Land of grandson:90

Process returned 0 (0x0)   execution time : 15.221 s
Press any key to continue.
```

Problem 5:

Inheritance Son(As Protected) & Grandson(As Protected):

```
#include<iostream>
using namespace std;
class father
{
   int money; protected:
   int gold; public:
int land;     int
set_money()
```

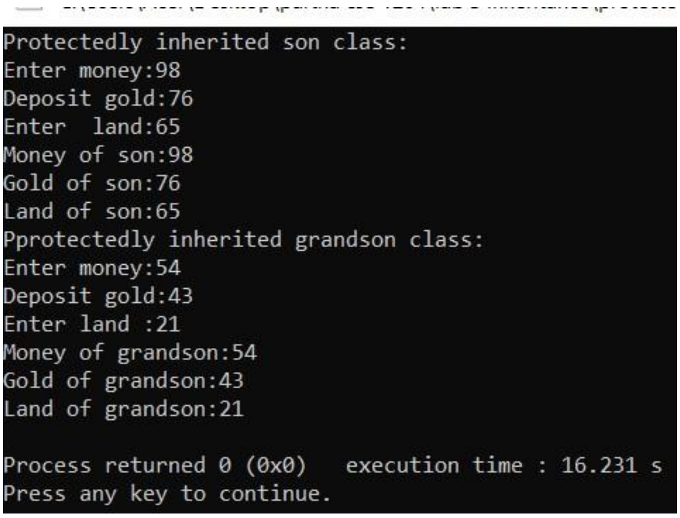
```
   {
      cout <<"Enter money:";
cin>>money;
   }
   int get_money()
   {
      return money;
   }
};
class son : protected   father
{
public:     int
get_money_son()
   {
      get_money();
   }
   int set_money_son()
   {
      set_money();
   }
   int x;
   void set_gold_son()
   {
      cout<<"Deposit gold:";
      cin >>x;
      gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
   void set_land_son()

{        cin >>
x;
      land=x;
   }
   int get_land_son()
   {
      return land;
   }
};
class grandson: protected son
{
public:     int
set_money_gson()
   {
      set_money_son();
```

```
  }
  int get_money_gson()
  {
    get_money_son();
  }
  int set_gold_gson()
  {
    set_gold_son();
  }
  int get_gold_gson()
  {
    get_gold_son();
  }
  int set_land_gson()
  {
    set_land_son();
  }
  int get_land_gson()
  {
    get_land_son();
  }
};
int main()
{    son ob;
  cout<<"Protectedly inherited son class:\n";
ob.set_money_son();        ob.set_gold_son();
cout<<"Enter  land:";    ob.set_land_son();
  cout<<"Money of son:"<<ob.get_money_son()<<endl;
cout<<"Gold of son:"<<ob.get_gold_son()<<endl;    cout<<"Land
of son:"<<ob.get_land_son()<<endl;    cout<<"Protectedly
inherited grandson class:\n";
  grandson ob_gson;
ob_gson.set_money_gson() ;
ob_gson.set_gold_gson() ;
cout<<"Enter land :";
ob_gson.set_land_gson() ;
  cout<<"Money of grandson:"<<ob_gson.get_money_gson() <<endl;
cout<<"Gold of grandson:"<<ob_gson.get_gold_gson() <<endl;
cout<<"Land of grandson:"<<ob_gson.get_land_gson() <<endl;    return 0;
}
```

Discussion :

Father: Need function to give access of money.
Son :   Need function to access gold,land.
Grandson: Need function to access function of gold,money,land of son.

Output:



```
Protectedly inherited son class:
Enter money:98
Deposit gold:76
Enter  land:65
Money of son:98
Gold of son:76
Land of son:65
Pprotectedly inherited grandson class:
Enter money:54
Deposit gold:43
Enter land :21
Money of grandson:54
Gold of grandson:43
Land of grandson:21

Process returned 0 (0x0)   execution time : 16.231 s
Press any key to continue.
```

Problem 6:

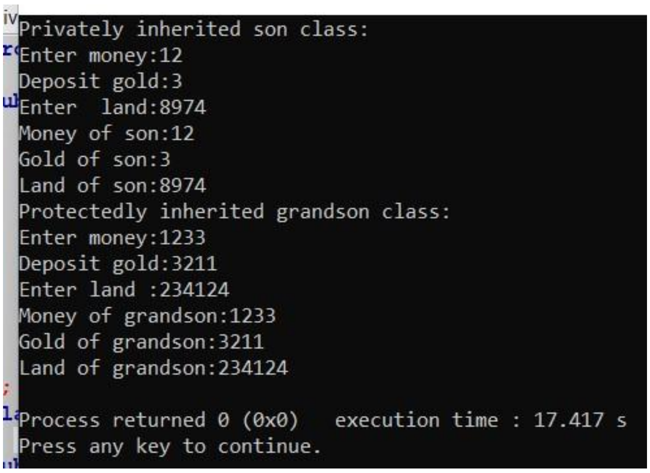
Inheritance Son(Privately) & Grandson(As Protected):

```
#include<iostream>
 using namespace std;
 class father
{
   int money; protected:
   int gold;
public:
int land;
   void set_money()
   {
      cout <<"Enter money:";
cin>>money;
   }
   int get_money()
   {
      return money;
   }
};
class son : private  father
{
public:    int
get_money_son()
   {
      get_money();
   }
   int set_money_son()
   {
      set_money();
   }
   int x;
   void set_gold_son()
   {
      cout<<"Deposit  gold:";
cin >>x;        gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
   void set_land_son()

{        cin >>
x;
land=x;
   }
   int get_land_son()
   {
      return land;
   }
};
class gson: protected son
{
public:     int
get_money_gson()
   {
      get_money_son();
   }
   int set_money_gson()
   {
      set_money_son();
   }

   void set_gold_gson()
   {
      set_gold_son();

   }
   int get_gold_gson()
   {
      get_gold_son();
   }
   void set_land_gson()
   {

      set_land_son();
   }
   int get_land_gson()
   {
      get_land_son();
   }
};
int main()
{     son ob;
   cout<<"Privately inherited son class:\n";
   ob.set_money_son();
ob.set_gold_son();     cout<<"Enter
land:";     ob.set_land_son();
   cout<<"Money of son:"<<ob.get_money_son()<<endl;
cout<<"Gold of son:"<<ob.get_gold_son()<<endl;     cout<<"Land
of son:"<<ob.get_land_son()<<endl;     cout<<"Protectedly
inherited grandson class:\n";
   grandson ob_gson;     ob_gson.set_money_gson();
ob_gson.set_gold_gson();     cout<<"Enter land :";
ob_gson.set_land_gson();     cout<<"Money of
grandson:"<<ob_gson.get_money_gson()<<endl;     cout<<"Gold of
grandson:"<<ob_gson.get_gold_gson()<<endl;     cout<<"Land of
grandson:"<<ob_gson.get_land_gson()<<endl;

   return 0;
}
```

Discussion :

Father: Need function to give access of money.
Son : Need function to access gold,land.
Grandson: Need function to access function of gold,money,land of son.

Output:



```
Privately inherited son class:
Enter money:12
Deposit gold:3
Enter  land:8974
Money of son:12
Gold of son:3
Land of son:8974
Protectedly inherited grandson class:
Enter money:1233
Deposit gold:3211
Enter land :234124
Money of grandson:1233
Gold of grandson:3211
Land of grandson:234124

Process returned 0 (0x0)   execution time : 17.417 s
Press any key to continue.
```

Problem 7:

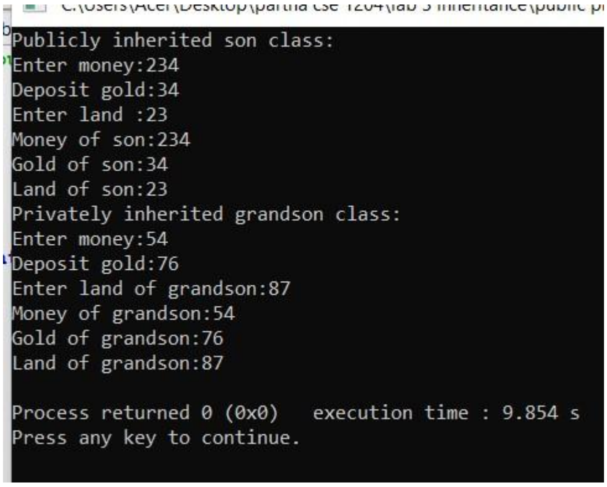
Inheritance Son(Publicly) & Grandson(Privately):

```
#include<iostream>
 using namespace std;
class father
{
   int money; protected:
   int gold; public:
int land;     void
set_money()
   {
   cout <<"Enter money:";
   cin>>money;
   }
   int get_money(){
   return money;
   }
};
class son : public   father
{
public:
   int x;
   void set_gold_son()
   {
      cout<<"Deposit gold:";
      cin >>x;
      gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
};
class grandson: private son
{
public:

    void set_money_gson()
   {
  set_money();
   }
   int get_money_gson(){
   get_money();
   }
    void set_gold_gson()
   {
      set_gold_son();

   }
   int get_gold_gson()
   {
    get_gold_son();
   }
   void set_land()
   {
cout<<"Enter land of grandson:";
   cin>>land;
   }
   int get_land(){
   return land;
   }
};
int main()
{     son ob;
   cout<<"Publicly inherited son class:\n";
ob.set_money();     ob.set_gold_son();
cout<<"Enter land :";
   cin>>ob.land;
   cout<<"Money of son:"<<ob.get_money()<<endl;
cout<<"Gold of son:"<<ob.get_gold_son()<<endl;
cout<<"Land of son:"<<ob.land<<endl;     cout<<"Privately
inherited grandson class:\n";
   nati ob_gson;
ob_gson.set_money_gson();
ob_gson.set_gold_gson();
   ob_gson.set_land();
   cout<<"Money of grandson:"<<ob_gson.get_money_gson()<<endl;
cout<<"Gold of grandson:"<<ob_gson.get_gold_gson()<<endl;
   cout<<"Land of grandson:"<<ob_gson.get_land()<<endl;
return 0;
}
```

Discussion :

Father: Need function to give access of money.
Son : Need function to access gold.
Grandson: Need function to access function of gold of son. Need function to access function of money of father. Need function to access land.

Output:



```
Publicly inherited son class:
Enter money:234
Deposit gold:34
Enter land :23
Money of son:234
Gold of son:34
Land of son:23
Privately inherited grandson class:
Enter money:54
Deposit gold:76
Enter land of grandson:87
Money of grandson:54
Gold of grandson:76
Land of grandson:87

Process returned 0 (0x0)   execution time : 9.854 s
Press any key to continue.
```

Problem 8:

Inheritance Son(As Protected) & Grandson(Privately):

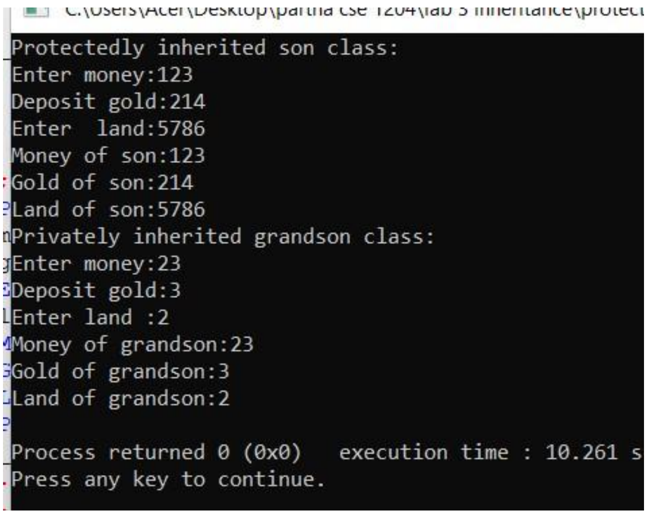
```
#include<iostream>
using namespace std;
class father
{
   int money; protected:
   int gold; public:
   int land;
   int set_money()
   {
      cout <<"Enter money:";
cin>>money;
   }
   int get_money()
   {
      return money;
   }
};
class son : protected   father
{
public:    int
get_money_son()
   {
      get_money();
   }
   int set_money_son()
   {
      set_money();
   }
   int x;
   void set_gold_son()
   {
        cout<<"Deposit  gold:";
cin >>x;        gold=x;
    }
    int get_gold_son()
    {
        return gold;
    }
    void set_land_son()

{        cin >>
x;
land=x;
    }
    int get_land_son()
    {
        return land;
    }
};
class grandson: private son
{
public:     int
set_money_gson()
    {
        set_money_son();
    }
    int get_money_gson()
    {
        get_money_son();
    }
    int set_gold_gson()
    {
        set_gold_son();
    }
    int get_gold_gson()
    {
        get_gold_son();
    }
    int set_land_gson()
    {
        set_land_son();
    }
    int get_land_gson()
    {
        get_land_son();
    }
};
int main()
{     son ob;
```

```
  cout<<"Protectedly inherited son class:\n";
ob.set_money_son();       ob.set_gold_son();
cout<<"Enter  land:";    ob.set_land_son();
  cout<<"Money of son:"<<ob.get_money_son()<<endl;
cout<<"Gold of son:"<<ob.get_gold_son()<<endl;
cout<<"Land of son:"<<ob.get_land_son()<<endl;
cout<<"Privately inherited grandson class:\n";
grandson ob_gson;    ob_gson.set_money_gson() ;
ob_gson.set_gold_gson() ;    cout<<"Enter land :";
ob_gson.set_land_gson() ;
  cout<<"Money of grandson:"<<ob_gson.get_money_gson() <<endl;
cout<<"Gold of grandson:"<<ob_gson.get_gold_gson() <<endl;
cout<<"Land of grandson:"<<ob_gson.get_land_gson() <<endl;
return 0;
}
```

Discussion :

Father **:** Need function to give access of money.
Son : Need function to access gold ,land. Need function to access function of money of father.
Grandson: Need function to access function of gold , money, land of son.

Output:



```
C:\Users\Acer\Desktop\partha cse 1204\lab 5 inheritance\protect
Protectedly inherited son class:
Enter money:123
Deposit gold:214
Enter  land:5786
Money of son:123
Gold of son:214
Land of son:5786
Privately inherited grandson class:
Enter money:23
Deposit gold:3
Enter land :2
Money of grandson:23
Gold of grandson:3
Land of grandson:2

Process returned 0 (0x0)   execution time : 10.261 s
Press any key to continue.
```

Problem 9:

Inheritance Son (Privately) & Grandson((Privately):

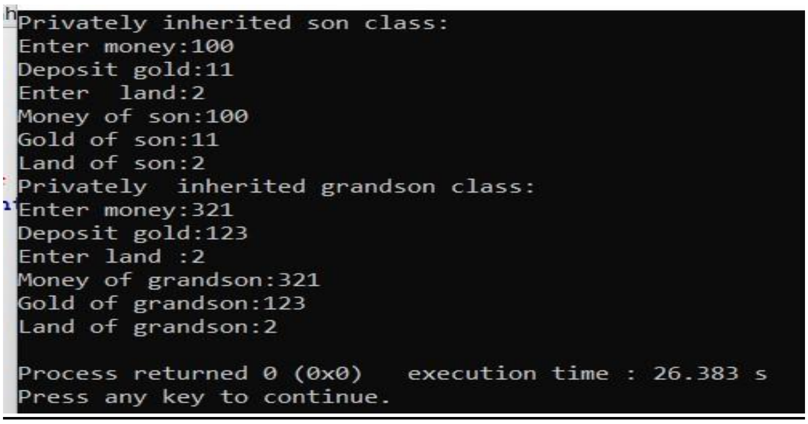
```
#include<iostream>
using namespace std;
class father
{
   int money; protected:
   int gold;
public:
int land;
   void set_money()
   {
      cout <<"Enter money:";
      cin>>money;
   }
   int get_money()
   {
      return money;
   }
};
class son : private  father
{
public:     int
get_money_son()
   {
      get_money();
   }
   int set_money_son()
   {
      set_money();
   }
   int x;
   void set_gold_son()
   {
      cout<<"Deposit gold:";
      cin >>x;
      gold=x;
   }
   int get_gold_son()
   {
      return gold;
   }
   void set_land_son()

{        cin >>
x;
      land=x;
   }
   int get_land_son()
   {
      return land;
   }
};
class grandson: private son
```

```
{
public:     int
get_money_gson()
   {
      get_money_son();
   }
   int set_money_gson()
   {
      set_money_son();
   }
   void set_gold_gson()
   {
      set_gold_son();
   }
   int get_gold_gson()
   {
      get_gold_son();
   }
   void set_land_gson()
   {
      set_land_son();
   }
   int get_land_gson()
   {
      get_land_son();
   }
};
int main()
{     son ob;
   cout<<"Privately inherited son class:\n";
   ob.set_money_son();
ob.set_gold_son();     cout<<"Enter
land:";     ob.set_land_son();
   cout<<"Money of son:"<<ob.get_money_son()<<endl;
cout<<"Gold of son:"<<ob.get_gold_son()<<endl;     cout<<"Land
of son:"<<ob.get_land_son()<<endl;     cout<<"Privately
inherited grandson class:\n";
   grandson ob_gson;
ob_gson.set_money_gson();
ob_gson.set_gold_gson();
cout<<"Enter land :";
ob_gson.set_land_gson();
   cout<<"Money of grandson:"<<ob_gson.get_money_gson()<<endl;     cout<<"Gold of
grandson:"<<ob_gson.get_gold_gson()<<endl;
   cout<<"Land of
grandson:"<<ob_gson.get_land_gson()<<endl;     return 0; }
```

Discussion :

Father : Need function to give access of money.
Son : Need function to access function of gold ,land of son. Need function to access function of money of father.
Grandson: Need function to access function of gold , money, land of son.

Output:



```
Privately inherited son class:
Enter money:100
Deposit gold:11
Enter  land:2
Money of son:100
Gold of son:11
Land of son:2
Privately  inherited grandson class:
Enter money:321
Deposit gold:123
Enter land :2
Money of grandson:321
Gold of grandson:123
Land of grandson:2

Process returned 0 (0x0)   execution time : 26.383 s
Press any key to continue.
```