Class and object practical some example

1.

```
#include<iostream>
#include<conio.h>
using namespace std;
class square
{   public:
    int length;
    int finalarea;
    void getlength()
    {
        cout<<"enter the length of square";
        cin>>length;

    }
    void square_area()
    {
        finalarea=length*length;
        cout<<"the final area of square is " ""<<finalarea<<endl;
    }
};
int main()
{
 square takelength;
 takelength.getlength();
 takelength.square_area();
 return 0;
}
```

2.

```
#include<iostream>
#include<conio.h>
using namespace std;
class name
{ public:
  char name[20];
  void getname()
  {
    cout<<"enter name";
    cin>>name;
    cout<<"the name is"<<name;
  }
};
int main()
{  // creating object of class
    name n1;
    n1.getname();
    return 0;
}
```

3.

```
#include<iostream>
#include<conio.h>
using namespace std;
class hello
{   public:
    char myname[20]="sunny";
    void show()
    {
        cout<<"hello world"<<myname;
    }
};
int main()
{
   hello hii;
   hii.show();
   return 0;
}
```

2.consrructor and destructor inheritance

```
#include<iostream>
using namespace std;
class a
{  public:
    a()
    {
        cout<<"first class object"<<endl;
    }
    ~a()
    {
        cout<<"first class object is destroyed"<<endl;
    }
};
class b:public a
{
    public:
     b()
    {
        cout<<"second class object"<<endl;
    }
    ~b()
    {
        cout<<"second class object is destroyed"<<endl;
    }

};
int main()
{
    b all;
    return 0;
}
```

3.destructor.cpp

```cpp
#include<iostream>
#include<conio.h>
using namespace std;
class b2k
{ public:
  b2k()
  {
    cout<<"object is born"<<endl;
  }
  ~b2k()
  {
    cout<<"object is killed or destroy"<<endl;
  }
};
int main()
{
    b2k obj1;
    return 0;
}
```

4.

```cpp
#include<iostream>
#include<conio.h>
using namespace std;
inline void show();
int main()
{
    show();
    return 0;
}
inline void show()
{
    cout<<"hello its deepak i love you ♡";
}
```

5.Parameterixesd constructor

```
1. #include<iostream>
#include<conio.h>
using namespace std;
class para
{
int a;
int b;
public:
para(int x,int y)
{
  a=x;
  b=y;
}
void show()
{
   int c;
   c=a+b;
   cout<<"the addition of two number is"<<c<<endl;
   cout<<"the number are a="<<a<<endl<<"the number b="<<b;

}
};
int main()
{
    para takename(190,20);
    takename.show();
    return 0;
}
```

2.parameterized

```
#include<iostream>
#include<conio.h>
using namespace std;
class para{
     char single;
     public:
     para(char siglefromuser )
     {
       single=siglefromuser;
       cout<<"the single character from user"<<single;
     }
};
int main()
{
   para h('p');
   return 0;
}
```

3.

3.
```cpp
#include<iostream>
#include<conio.h>
using namespace std;
class para
{
   int a;
   public:
   para(int x)
   {
    a=x;
   }
   void show()
   {
    cout<<"the number is a="<<a;
   }
};
int main()
{
    para b(420);
    b.show();
    return 0;
}
```

6.single inheritance

1.
```cpp
#include<iostream>

#include<conio.h>
using namespace std;
class squarelength
{   public:
    float length;
    void getlength()
    {
        cout<<"enter the length of square";
        cin>>length;
    }
};
class squrearea:public squarelength
{  public:
   float finalarea;
   void calcarea()
   {
    finalarea=length*length;
    cout<<"the area of square is"<<finalarea;
   }
};
int main()
{
    squrearea s1;
    s1.getlength();
    s1.calcarea();
    return 0;
}
```

2.
```cpp
#include<iostream>

#include<conio.h>
using namespace std;
class number
{   public:
    int a=60;
};
class numbertype:public number
{   public:
    void get()
    {
        cout<<"the number from base class"<<a;
    }
};
int main()
{
    numbertype num;
    num.get();
    return 0;
}
```

3.
```cpp
#include<iostream>

#include<conio.h>
using namespace std;
class base
{
  public:
  void show()
  {
    cout<<"hello from parent class";
  }
};
class derieve:public base
{   // class inherited
    public:
};
int main()
{
    derieve d1;
    d1.show();
    return 0;
}
```