```
1  //============================================================================
2  // Name        : Animal.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is an abstract class contains: virtual destructor, count of
     all object created from Animal, operator<< overload,...etc.
6  //============================================================================
7  #ifndef ANIMAL_H_
8  #define ANIMAL_H_
9  #include <string>
10 class Animal
11 {
12 public:
13     virtual std::string talk() = 0;
14     //Precondition:
15     //  _None.
16     //Postcondition:
17     //  _Virtual method used to initilize how the Animal's talk.
18
19     virtual std::string move() = 0;
20     //Precondition:
21     //  _None.
22     //Postcondition:
23     //  _Virtual method used to initilize how the Animal's move.
24
25     virtual ~Animal() = 0;
26     //Precondition:
27     //  _None.
28     //Postcondition:
29     //  _Virtual destructor used to deallocated the memory used to initilize
         variable.
30
31     std::string getAnimalType();
32     //Precondition:
33     //  _None.
34     //Postcondition:
35     //  _Return the name of this animal.
36
37     int getCount();
38     //Precondition:
39     //  _None.
40     //Postcondition:
41     //  _Return the number of this object existed.
42
43     friend std::ostream& operator<<(std::ostream& os, Animal& obj);
44     //Precondition:
45     //  _None.
46     //Postcondition:
47     //  _Return os contains animalType, animal's talk, animal's move.
48 protected:
49     std::string *animalType = NULL;
50     //Precondition:
```

```
51      //  _None.
52      //Postcondition:
53      //  _This pointer pointed to a animalType string.
54
55      static int count;
56      //Precondition:
57      //  _None.
58      //Postcondition:
59      //  _This varaible created on the number of Animal existed.
60
61  };
62  #endif
63
64
```

```cpp
//============================================================================
// Name        : Animal.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is an abstract class contains: virtual destructor, count of all object created from Animal, operator<< overload,...etc.
//============================================================================

#include "Animal.h"
#include<iostream>

//Precondition:
//  _None.
//Postcondition:
//  _This varaible created on the number of Animal existed.
int Animal::count = 0;

//Precondition:
//  _None.
//Postcondition:
//  _Virtual destructor used to deallocated the memory used to initilize variable.
Animal::~Animal()
{
    delete animalType;
    count--;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return the name of this animal.
std::string Animal::getAnimalType()
{
    return "["+*animalType+"]";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return the number of this object existed.
int Animal::getCount()
{
    return count;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return os contains animalType, animal's talk, animal's move.
std::ostream & operator<<(std::ostream & os,  Animal & obj)
{
    os << *(obj.animalType) << ", " << obj.talk() << ", " << obj.move();
```

```
52      return os;
53  }
54
```

```
1  //============================================================================
2  // Name        : bird.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a derived from class Animal.
6  //============================================================================
7  #ifndef BIRD_H_
8  #define BIRD_H_
9  #include "Animal.h"
10 class bird:public Animal
11 {
12 public:
13     bird();
14     //Precondition:
15     //  _None.
16     //Postcondition:
17     //  _None.
18 };
19 #endif
20
```

```cpp
//============================================================================
// Name        : bird.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a derived from class Animal.
//============================================================================

#include "bird.h"

//Precondition:
//  _None.
//Postcondition:
//  _None.
bird::bird()
{

}
```

```
1  //============================================================================
2  // Name        : mammal.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a derived from class Animal.
6  //============================================================================
7  #ifndef MAMMAL_H_
8  #define MAMMAL_H_
9  #include "Animal.h"
10 class mammal:public Animal
11 {
12 public:
13     mammal();
14     //Precondition:
15     //  _None.
16     //Postcondition:
17     //  _None.
18 };
19 #endif
20
```

```cpp
//============================================================================
// Name        : mammal.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a derived from class Animal.
//============================================================================

#include "mammal.h"

//Precondition:
//  _None.
//Postcondition:
//  _None.
mammal::mammal()
{

}
```

```
1  //============================================================================
2  // Name        : repitle.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a derived from class Animal.
6  //============================================================================
7  #ifndef REPITLE_H_
8  #define REPITLE_H_
9  #include "Animal.h"
10 class repitle: public Animal
11 {
12 public:
13     repitle();
14     //Precondition:
15     //  _None.
16     //Postcondition:
17     //  _None.
18 };
19
20
21 #endif
```

```cpp
//============================================================================
// Name        : repitle.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a derived from class Animal.
//============================================================================

#include "repitle.h"

//Precondition:
//  _None.
//Postcondition:
//  _None.
repitle::repitle()
{
}
```

```
1  //==========================================================================
2  // Name        : chicken.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class bird which implement all the
     neccessary constructor and methods.
6  //==========================================================================
7  #ifndef CHICKEN_H_
8  #define CHICKEN_H_
9  #include <string>
10 #include "bird.h"
11 class chicken:public bird
12 {
13 public:
14     chicken();
15     //Precondition:
16     //  _None.
17     //Postcondition:
18     //  _Initilize the animalType to "chicken".
19
20     std::string talk();
21     //Precondition:
22     //  _None.
23     //Postcondition:
24     //  _Return chicken's talk.
25
26     std::string move();
27     //Precondition:
28     //  _None.
29     //Postcondition:
30     //  _Return chicken's move.
31 };
32 #endif
33
```

```cpp
//============================================================================
// Name        : chicken.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class bird which implement all the neccessary constructor and methods.
//============================================================================

#include "chicken.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "chicken".
chicken::chicken()
{
    animalType = new std::string("chicken");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return chicken's talk.
std::string chicken::talk()
{
    return "crow";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return chicken's move.
std::string chicken::move()
{
    return "walk";
}


```

```
1  //==========================================================================
2  // Name        : eagle.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class bird which implement all the
     neccessary constructor and methods.
6  //==========================================================================
7  #ifndef EAGLE_H_
8  #define EAGLE_H_
9  #include <string>
10 #include "bird.h"
11 class eagle:public bird
12 {
13 public:
14     eagle();
15     //Precondition:
16     //  _None.
17     //Postcondition:
18     //  _Initilize the animalType to "eagle".
19
20     std::string talk();
21     //Precondition:
22     //  _None.
23     //Postcondition:
24     //  _Return eagle's talk.
25
26     std::string move();
27     //Precondition:
28     //  _None.
29     //Postcondition:
30     //  _Return eagle's move.
31 };
32 #endif
33
34
```

```cpp
//============================================================================
// Name        : eagle.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class bird which implement all the neccessary constructor and methods.
//============================================================================

#include "eagle.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "eagle".
eagle::eagle()
{
    animalType = new std::string("eagle");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return eagle's talk.
std::string eagle::talk()
{
    return "call";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return eagle's move.
std::string eagle::move()
{
    return "fly";
}


```

```
1  //==========================================================================
2  // Name        : bear.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class mammal which implement all
     the neccessary constructor and methods.
6  //==========================================================================
7  #ifndef BEAR_H_
8  #define BEAR_H_
9  #include <string>
10 #include "mammal.h"
11
12 class bear:public mammal
13 {
14 public:
15     bear();
16     //Precondition:
17     //  _None.
18     //Postcondition:
19     //  _Initilize the animalType to "bear".
20
21     std::string talk();
22     //Precondition:
23     //  _None.
24     //Postcondition:
25     //  _Return bear's talk.
26
27     std::string move();
28     //Precondition:
29     //  _None.
30     //Postcondition:
31     //  _Return bear's move.
32
33 };
34 #endif
35
36
```

```cpp
//============================================================================
// Name        : bear.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class mammal which implement all the neccessary constructor and methods.
//============================================================================
#include "bear.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "bear".
bear::bear()
{
    animalType = new std::string("bear");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return bear's talk.
std::string bear::talk()
{
    return "growl";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return bear's move.
std::string bear::move()
{
    return "walk";
}


```

```
1  //==========================================================================
2  // Name        : hyena.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class mammal which implement all
     the neccessary constructor and methods.
6  //==========================================================================
7  #ifndef HYENA_H_
8  #define HYENA_H_
9  #include <string>
10 #include "mammal.h"
11
12 class hyena:public mammal
13 {
14 public:
15     hyena();
16     //Precondition:
17     //  _None.
18     //Postcondition:
19     //  _Initilize the animalType to "hyena".
20
21     std::string talk();
22     //Precondition:
23     //  _None.
24     //Postcondition:
25     //  _Return hyena's talk.
26
27     std::string move();
28     //Precondition:
29     //  _None.
30     //Postcondition:
31     //  _Return hyena's move.
32 };
33 #endif
34
35
```

```cpp
// Name        : hyena.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class mammal which implement all the neccessary constructor and methods.
//==============================================================================

#include "hyena.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "hyena".
hyena::hyena()
{
    animalType = new std::string("hyena");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return hyena's talk.
std::string hyena::talk()
{
    return "howl";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return hyena's move.
std::string hyena::move()
{
    return "run";
}


```

```
1  //==========================================================================
2  // Name        : lion.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class mammal which implement all
     the neccessary constructor and methods.
6  //==========================================================================
7  #ifndef LION_H_
8  #define LION_H_
9  #include <string>
10 #include "mammal.h"
11 class lion:public mammal
12 {
13 public:
14     lion();
15     //Precondition:
16     //  _None.
17     //Postcondition:
18     //  _Initilize the animalType to "lion".
19
20     std::string talk();
21     //Precondition:
22     //  _None.
23     //Postcondition:
24     //  _Return lion's talk.
25
26     std::string move();
27     //Precondition:
28     //  _None.
29     //Postcondition:
30     //  _Return lion's move.
31 };
32 #endif
33
34
```

```cpp
//============================================================================
// Name        : lion.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class mammal which implement all the neccessary constructor and methods.
//============================================================================

#include "lion.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "lion".
lion::lion()
{
    animalType = new std::string("lion");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return lion's talk.
std::string lion::talk()
{
    return "roar";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return lion's move.
std::string lion::move()
{
    return "run";
}


```

```
1  //============================================================================
2  // Name        : lizard.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class repitle which implement all
     the neccessary constructor and methods.
6  //============================================================================
7  #ifndef LIZARD_H_
8  #define LIZARD_H_
9  #include <string>
10 #include "repitle.h"
11 class lizard:public repitle
12 {
13 public:
14     lizard();
15     //Precondition:
16     //  _None.
17     //Postcondition:
18     //  _Initilize the animalType to "lizard".
19
20     std::string talk();
21     //Precondition:
22     //  _None.
23     //Postcondition:
24     //  _Return lizard's talk.
25
26     std::string move();
27     //Precondition:
28     //  _None.
29     //Postcondition:
30     //  _Return lizard's move.
31 };
32 #endif
33
```

```cpp
//============================================================================
// Name        : lizard.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class repitle which implement all the neccessary constructor and methods.
//============================================================================

#include "lizard.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "lizard".
lizard::lizard()
{
    animalType = new std::string("lizard");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return lizard's talk.
std::string lizard::talk()
{
    return "cry";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return lizard's move.
std::string lizard::move()
{
    return "crawl";
}


```

```
1  //============================================================================
2  // Name        : snake.h
3  // Author      : Sotheanith Sok
4  // Version     : 1.0
5  // Description : This is a concrete class from class repitle which implement all
     the neccessary constructor and methods.
6  //============================================================================
7  #ifndef SNAKE_H_
8  #define SNAKE_H_
9  #include <string>
10 #include "repitle.h"
11 class snake:public repitle
12 {
13 public:
14     snake();
15     //Precondition:
16     //  _None.
17     //Postcondition:
18     //  _Initilize the animalType to "snake".
19
20     std::string talk();
21     //Precondition:
22     //  _None.
23     //Postcondition:
24     //  _Return snake's talk.
25
26     std::string move();
27     //Precondition:
28     //  _None.
29     //Postcondition:
30     //  _Return snake's move.
31 };
32 #endif
33
34
```

```cpp
//============================================================================
// Name        : snake.cpp
// Author      : Sotheanith Sok
// Version     : 1.0
// Description : This is a concrete class from class repitle which implement all the neccessary constructor and methods.
//============================================================================

#include "snake.h"

//Precondition:
//  _None.
//Postcondition:
//  _Initilize the animalType to "snake".
snake::snake()
{
    animalType = new std::string("snake");
    count++;
}

//Precondition:
//  _None.
//Postcondition:
//  _Return snake's talk.
std::string snake::talk()
{
    return "hisses";
}

//Precondition:
//  _None.
//Postcondition:
//  _Return snake's move.
std::string snake::move()
{
    return "crawl";
}


```