

```
finn@BMO:~$ cowsay -n ice-dragon Ice-cold RAWR
```

Ice-cold RAWR

```
      /\_/\
     /0  0 \
    / ^__^ \
   //  _\\//
  (  __\ (
 (  ___(
 (  ___(
 ((  ___)) ,--{
((  ___))  '\
((  ___))   {             }
((  ___))   .-----.\       \'
              ///.-----..>         \
              ///-.-.-.-.-}^-----~
```

This dragon cannot breathe fire.

Cow Class

The **Cow** class must have all the same methods as previously required, though students may add private methods. The methods are repeated here, briefly, for reference. (Note that **setImage()** should be marked as virtual.)

```
public Cow(const string& _name)           // Constructor
public string& getName()                  // Returns name of this cow object
public string& getImage()                 // Return image for this cow object
public virtual void setImage(const string& _image) // Sets image for this cow object
```

Dragon Class

The **Dragon** class must be derived from the **Cow** class and must make all its methods available. You may mark any additional methods virtual as necessary. In addition, Dragon must provide the following methods:

```
public Dragon(const string& _name, const string& _image)
```

Constructor; creates new **Dragon** object with given **name** and **image**. This must be its **only** public constructor!

```
public bool canBreatheFire()
```

This method should exist in **every** **Dragon** class. For the default **Dragon** type, it should always return **true**.

IceDragon Class

The **IceDragon** class must be derived from the **Dragon** class and must make all its methods available:

```
public IceDragon(const string& _name, const string& _image)
```

Constructor; creates a new **IceDragon** object with the given **name** and **image**. This should be the **only** public constructor for the **IceDragon** class!

```
public bool canBreatheFire()
```

For the **IceDragon** type, this method should always return **false**.

HeiferGenerator Class (Provided)

```
public static vector<Cow*>& getCows()
```

Returns a reference to a vector of cow object pointers from built-in data set. This will call the **Cow** constructor and **setImage()** methods of the cow class if needed to initialize new cow objects uniquely for each data set.

```
public static Dragon* getDragonPointer(Cow* candidate)
```

If object pointed to by **candidate** is a **Dragon** (including an **IceDragon**), returns a **Dragon** pointer to the object. Otherwise, returns **nullptr**.

Submissions

NOTE: Your output must match the example output **exactly**. If it does not, ***you will not receive full credit for your submission!*** (Note that matching sample output is necessary, but not sufficient, for full credit.)

Files: cowsay.cpp, Cow.cpp, Cow.h, Dragon.cpp, Dragon.h, IceDragon.cpp, IceDragon.h, CMakeLists.txt

Method: Submit on Canvas

Sample Output

```
finn@BMO:~$ ./cowsay
finn@BMO:~$ ./cowsay Hello World!
```

```
Hello World!
```

