Chapter 7 Objects and Classes

1. See the section "Defining Classes for Objects."

2. Define the initializer, create data fields, and define methods.

3. Use a constructor

4. The name of the initializer is __init__.

5. The self refers to the object itself. Through self, the members of the object can be accessed.

6. The syntax for constructing an object is

   ClassName(arguments)

   The arguments of the constructor match the parameters in the __init__ method without self.

   The constructor first creates an object in the memory and then invokes the initializer.

7. Initializer is a special method that is called when creating an object.

8. The object member access operator is the dot (.).

9. You need to pass an argument in the constructor A() to invoke the class A's initializer.

10. (a) The constructor should be defined as __init__(self).

    (b) radius = 3 should be self.radius = 3

11.
```
count is 100
times is 0
```

12.
```
count is 0
n is 1
```

13. __i is a private data field and cannot be accessed from outside of the class.

14. Correct. The printout is Welcome.

15. __on is a private data field and cannot be accessed outside the class.

    The way to fix it is to add a getter method for the Boolean property as follows:

```
class A:
    def __init__(self, on):
        self.__on = not on

    def isOn(self):
        return self.__on

def main():
    a = A(False)
    print(a.isOn())

main() # Call the main function
```

16. Two benefits: (1) for protecting data and (2) for easy to maintain the class.
    In Python, private data fields are defined with two leading underscores.

17. Add two underscores as the prefix for the method name.
18. See the text