Name: Minjie Shen

CS 5004 Module 1 Worksheet
Data Definitions, Classes & Testing

Remember & Understand

1. Consider the following program. Write out the line of code that would get this program to print out the message "I Rule!". Where would you put it?

```
public class MyFirstProgram {


  public static void main(String[] args) {

System.out.println(“I Rule!”);
  }


}
```

2. What is a variable?

Variable in Java is a data container that saves the data values during Java program execution. Every variable is assigned a data type that designates the type and quantity of value it can hold. For example, String is a data type, and String a = “a”; a is a variable here.

3. What is the difference between a *statically-typed* and a *dynamically-typed* language?

   In statically typed language, each variable and expression is already known at compile time. However, in dynamically-typed language, variables can receive different values at runtime and their type is defined at run time.

4. Why would a *statically-typed* language be a **good thing** when you are writing large programs?

When writing large programs, statically typed language provides a kind of protection to data variables. Those data variables cannot be modified in different programs. So, it reduces the unexpected bugs for large programs.

5. For each of the following categories, indicate which one is used *most often* when writing Java programs:
   Answers are underlined.

Integral values:	byte	short	<u>int</u>	long
Decimal values:	float	<u>double</u>		

6. What two primitive types are missing from the previous question?

char data type and Boolean data type.

7. In Java, there is a different meaning between *single* (') and *double* (") quotes. When would you use one version the other?

   Single quote is used for char data type; double quote is used for String data type.

8. The declarations in a class can be written in any order, however best practices say that we should always put things in the same place. In this module, we have put the elements in the order that we expect to see them in your classes. Rewrite the following declarations in the order that they should appear in your file: (1) method, (2) field, (3) constructor, (4) class.

The order should be (4) class, (2) field, (3) constructor, and (1) method.

9. In a JUnit test class, what is the purpose of the method that is annotated with @Before?

The method annotated with @Before is executed before each method annotated with @Test.

10. What two characteristics does every JUnit test need to have to make it a test case?

Each Junit test needs to have @Test annotated method, and the assertEquals method.

11. How does a test method pass?

Normally, a test method passes an assertEquals method. The Test annotation tells JUnit that the public void method to which it is attached can be run as a test case. To run the method, JUnit first constructs a fresh instance of the class then invokes the annotated method. Any exceptions thrown by the test will be reported by JUnit as a failure. If no exceptions are thrown, the test is assumed to have succeeded.

12. What is the difference between the `public` and `private` access modifiers?

“Private” means this method or variable can only be accessed and modified within this class; However, “Public” means this method or variable can be accessed and modified outside of the class.

13. Given the following method, write a valid Javadoc comment for it.

```
/**
* this class runs like the operator “+”, and it takes in two int values, returns the sum of the two int
* variables.
*/
     int addition(int one, int two) {
       int sum;
       sum = one + two;
       return sum;
     }
```

14. Compare the following two versions of the same class diagram from Lesson 10:

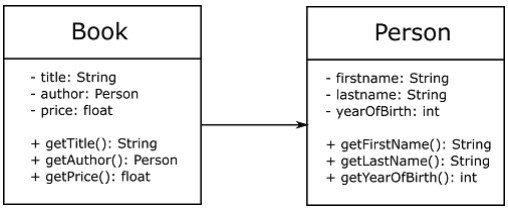
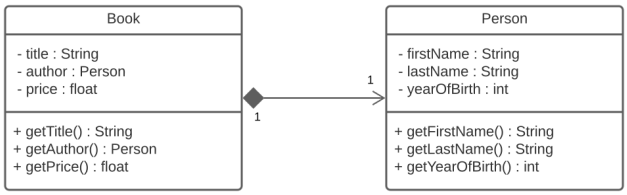


Diagram used in the video



Same diagram that follows the UML standard

What are three distinct differences between the two diagrams?

1) The UML standard diagram separates the private and public methods and variables into two blocks, while the diagram used in the video does not do that.
2) The UML standard diagram displays the arrow that represents Book class contains 1 Person object in it.
3) The UML standard diagram displays the numbers on the relationship between classes. The number 1 indicates that book class contains only 1 Person object.

Application

1. What is the output of the following fragment?

```
int quotient = 7 / 3;
int remainder = 7 % 3;
System.out.println("quotient = " + quotient);
System.out.println("remainder =" + remainder);
```

Output will be:

quotient = 2
remainder = 1

2. Given the following fragment that tries to convert Celsius to Fahrenheit:

```
double celsius = 20;
double fahrenheit;
fahrenheit = (9 / 5) * celsius + 32.0;
```

a. What value is assigned to fahrenheit?

   Fahrenheit is 52.0

b. Explain what is actually happening and what the programmer likely wanted.

   The actual value the programmer wants will be 68, which is 20*(9/5) + 32.0; the error here is because of the 9 / 5 are interpretated as two integer, and int 9 / 5 will be 1. In order to fix this problem, we need to convert the 9 and 5 to double data type, and write it like “`fahrenheit = (9.0 / 5.0) * celsius + 32.0;`” the error will be fixed.

c. Rewrite the code so that it does what the programmer intended

```
double celsius = 20;
double fahrenheit;
fahrenheit = (9.0 / 5.0) * celsius + 32.0;
```

3. Consider what we learned about the difference between blocks in Python and blocks in Java. Then look at the following Python code and rewrite it using Java (You may assume that `input` is a method that already exists. We will learn how to read from the keyboard in another lesson):

```python
def main():
    menu = '0 -- Quit\n1 -- Add "O"\n2 -- Add "oo"\n3 -- Add "xXx"'
    result = ""

    print(menu)
    choice = input("Option: ")

    while choice != 0:
        if choice == 1:
            result += "O"
        elif choice == 2:
            result += "oo"
        elif choice == 3:
            result += "xXx"
        else:
            print("Invalid option")
        print(menu)
        choice = input("Option: ")

    print("Result:", result)
```

```java
public class MyClass {
    private String menu = "0 -- Quit\n1 -- Add \"O\"\n2 -- Add \"oo\"\n3 -- Add \"xXx\"";
    private String result = "";

   public static void main(String args[]) {
      System.out.println(menu);
      int choice = input("Option: ");  /* assume the input method is already existed */
      while (choice != 0) {
         if(choice == 1) {
            result += "o";
         } else if(choice == 2) {
            result += "oo";
         } else if(choice == 3) {
            result += "xXx";
         } else {
            System.out.println("Invalid option " + menu);
            choice = input("Option: ");
         }
      }
      System.out.println("Result: " + result);
   }
   }
```

4. Consider the following code fragments. Rearrange them (adding whatever curly braces you need) to that make a working Java program that produces the following output:

a-b c-d

Code fragments:

```
if (x == 1) {
  System.out.print("d");
  x = x - 1;
}
```

```
if (x == 2) {
  System.out.print("b c");
}
```

```
class Shuffle {
  public static void main(String[] args) {
```

```
if (x > 2) {
  System.out.print("a");
}
```

```
int x = 3;
```

```
x = x - 1;
System.out.print("-");
```

```
while (x > 0) {
```

```
class Shuffle {
  public static void main(String[] args) {
```

```
int x = 3;
```

```
while (x > 0) {
```

```
if (x > 2) {
  System.out.print("a");
}
```

```
if (x == 2) {
  System.out.print("b c");
}
```

```
if (x == 1) {
  System.out.print("d");
  x = x - 1;
}
```

```
x = x - 1;
System.out.print("-");
}
}
```

5. Each of the following represent a complete source file. Your job is to play compiler and determine whether each of these files will compile. If they won't, how would you fix them?

`public static void main(String [] args) {`
`  int x = 5;`
`  while (x > 1) {`
`  x = x - 1;`
`  if (x < 3) {`
`  System.out.println("small x");`
`  }`
`  }`
`}`
	These lines will not compile since there is no puclic class to execute the code.

`public class Exercise {`
`  public static void main(String [] args) {`
`  int x = 1;`
`  while (x < 10) {`
`  if (x > 3) {`
`  System.out.println("big x");`
`  }`
`  }`
`  }`
`}`
	These lines will compile, but the code is wrong and will not produce any outcome. It will be stucked in this while loop and never come out, which is stack overflow.

`public class Exercise {`
`  int x = 5;`
`  while (x > 1) {`
`  x = x - 1;`
`  if (x < 3) {`
`  System.out.println("small x");`
`  }`
`  }`
`}`
	These lines will not compile since there is no main function here to execute the code.

6. Consider each of the following statements. For each statement, fill in the table with the ***all*** of the following words for which that statement is always true:

class methods object instance variable

I am compiled from a .java file	Class, methods
My instance variable values can be different from my buddy's values	Class, methods
I behave like a template	Class, methods,
I like to do stuff	Methods, object
I can have many methods	class
I represent *state*	object
I have behaviors	methods
I am located in objects	methods
I am used to create object instances	instance variable
My state can change	instance variable
I declare methods	class
I can change at runtime	object

7. Given the following class, add the missing access modifiers.

```
class Dog {
   private String breed;
   private int age;
   private String color;

   public void barking() {
   }

   public void hungry() {
   }

   public void sleeping() {
   }
}
```

8. Given the following class, draw an appropriate UML diagram.

```
public class Movie {
  private String title;
  private String genre;
  private int rating;

  public void playIt() {
    System.out.println("Playing the movie");
  }
}
```

The UML diagram based on the code above is as follow:

Movie
- title: String - genre: String - rating: int
+ playIt(): void

9. Which part of the class is missing from the class provided in the previous question?

There is no main function in the class provided in the previous question, so the system.out.println command will not be able to compile.