CISC 3120
C03: Primitives and References

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Recap and issues
  - Selection & iteration
- Primitives and references
- Assignments

What did we learn from BeerSong.java?

- Anatomy of a Java class
  - What goes in a Java source code file, what goes in a Java class, and what goes in a method?
  - What is the entry of a Java program?
- A few data types
- Identifiers
- Simple and compound statements
- A few flow controls
- Comment
- Java build-in classes (Java libraries)
- Coding style

Practice: *W02-0_09-05*

- Work in class and in team
- 10 minutes
- Submit the worksheet

Using Command Line Arguments

- public static void main(String[] args)
  - An array of String objects passed to the main method
- How do we use it?
  - Example: use it to change BeerSong's behavior.

Selection Structures

- Similar to C++
- The if statement
  - The if-then statement
  - The if-then-else statement
- The switch statement (discuss later)

If-Then

- Example

```
if ( isMoving ) {
    currentSpeed --;
}
```

If-Then-Else

- Example

```
if (testscore >= 90) {
    grade = 'A';
} else if (testscore >= 80) {
    grade = 'B';
} else if (testscore >= 70) {
    grade = 'C';
} else if (testscore >= 60) {
    grade = 'D';
} else {
    grade = 'F';
}
```

Iterations

- The while statement
- The do statement (discuss later)
- The for statement

The while Statement

- while (expression) statement
- Example
  - BeerSong.java

The for Statement

- The basic for statement
- The enhanced for statement

The basic for Statement

- Example 1

```
for (int i=99; i>=0; i--) {
        System.out.println(i + "bottles of beers on the wall");
}
```

- Example 2

```
// print out command line arguments
for (int i=0; i<args.length; i++) {
        System.out.println(args[i])
}
```

The enhanced for Statement

- Example

```
// print out command line arguments
for (String anArg: args) {
        System.out.println(anArg)
}
```

Questions?

- Flow controls
  - Selections
  - Iterations

Classes and Objects

- Divide an application into multiple classes
- Instantiate objects from classes
- Thinking: client & server

The Guess Game

- Generate a list random numbers. Have 3 players to make a guess. See who makes correct guess. More of a simulation.
- Divide the application into 3 classes
  - GameLauncher
  - GuessGame
  - Player

Instantiating Objects

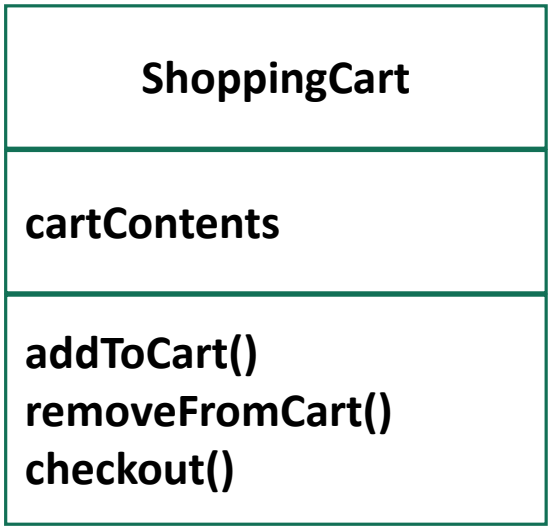
- Create objects from a class
  - Player player = new Player();

Calling Methods

- Client & server
- Within a class (whose object is a client):
  - player.guess()
  - player is the server

Describing a Class

- UML: Class diagram



Questions?

- objects

Primitive Data Types

- Java has 8 primitive data types

Type	Description	Default	Size	Example Literals
boolean	True or false	False	1 bit	true, false
byte	integer	0	8 bits	(none)
char	Unicode character	\u0000	16 bits	‘a’, ‘u0041’, ‘\101’
short	Integer	0	16 bits	(none)
int	Integer	0	32 bits	-9, -8, 0, 1 2
long	Integer	0	64 bits	3L, 1L, -1L, -3L
float	Floating point	0.0	32 bits	3.14e10f, -1.23e-100f
double	Floating point	0.0	64 bits	1.1e1d, -3.14e10d

References

- Other than primitives, everything else is an object in Java.
- Variables hold references to objects

The Dog Class

- Initialization
- Assignment
- Array of objects (references to the objects)

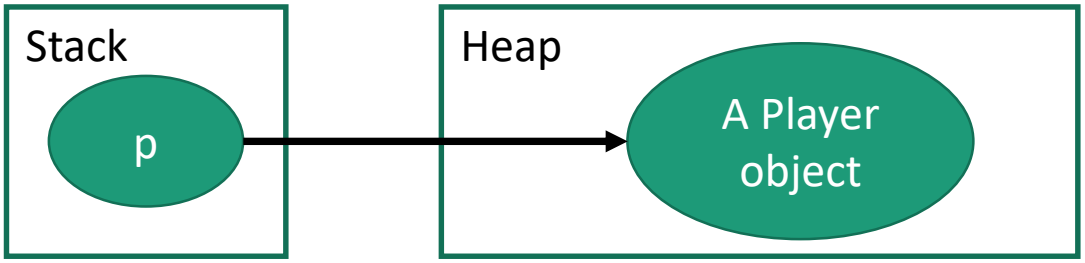
Where are the Objects?

- Player p = new Player();
- Compare with C++:
  - Where is the player object?
  - How do I "destroy" the object and release the memory?

Where are my Objects?

- JVM memory
  - Stack
    - Where local variables (a.k.a., stack variables) are allocated
  - (Garbage-Collection) Heap
    - Where objects are allocated (note: instance variables are part of an object)

```
public void startGame() {
    Player p = new Player();
}
```



Java Garbage Collector

- A program runs on the Java Virtual Machine (JVM)
  - Implements automatic memory management
  - Look for objects that are not being used by applications any more, and remove the objects, and freeing the memory.
- In Java, the garbage collector does the memory management for you.
- In C++, you needs to perform memory management (using the new and delete operators)

Questions

- Flow controls
  - Selection & iterations
- Classes and objects
- Primitive types and variables
- Objects and reference variables
- Use command line arguments
- JVM stack and garbage-collection heap

Assignments

- Practice examples to be posted at the class website