Introduction to Programming

with Java, for Beginners

++ & -- operator
char type
Main with arguments
Heap Management
Recap on static vs. non static

The increment operator
- ++ adds 1 to a variable
  - It can be used as a statement by itself, or within an expression
  - It can be put before or after a variable
  - If before a variable (pre-increment), it means to add one to the variable, then use the result
  - If put after a variable (post-increment), it means to use the current value of the variable, then add one to the variable
- The same applied to decrement operator

Examples of ++

int a = 5;
a++;
// a is now 6
int b = 5;
++b;
// b is now 6
int c = 5;
int d = ++c;
// c is 6, d is 6

Confusing code is bad code, so this is very poor style

char

- The primitive type char
  - Just stored as numbers
  - Each char as a unique integer value (based on Unicode standard)
- You can use characters in arithmetic (they will automatically be converted to int)
  - char ch = 'A';
  - ch + 1
    66
  - char ch2 = (char) (ch + 1) // cast result back to char B
Main

public static void main (String [] args)

- Must have the exact signature
  - Only variation allowed is name of the input parameter
- So main starts everything, how do we call main and provide inputs?

- To run a program recall
  - Command: `java ClassName`
  - This what calls the main method if the class has one
  - So we could pass arguments as follows:
    `java ClassName list-of-arguments`

Main with arguments example

public class ExampleArgs{
    public static void main(String [] args){
        System.out.println("Demo for Inputs args");
        for(int i = 0; i < args.length; i++){
            System.out.println(args[i]);
        }
    }
}

> java ExampleArgs ESE 112

Demo for Inputs args ESE 112

Note: Code works even if no arguments are passed to main() because JVM passes to main() a zero-length array of Strings and not a null

Example from Lab 6

public class MP3Player {
    public static void main(String[] args) {
        String filename = args[0];
        StdPlayer.open(filename);

        while (!StdPlayer.isEmpty()) {
            Wave w = new Wave(StdPlayer.getLeftChannel(), StdPlayer.getRightChannel());
            w.play();
        }

        StdPlayer.close();
        System.exit(0);
    }
}

Memory Management

- Memory is not infinite
- Stacks grow and shrink
- Heap
  - Grow when you dynamically allocate memory i.e. new Object()
  - If you do not manage the allocations then you will run out of this memory
    - Objects that will never be accessed or mutated again by application need to be reclaimed
    - This is known as Garbage Collection

- Some Languages like C/C++ leave it up to the programmer to do explicit memory management
- Java does automatic garbage collection
  - Done by JVM (Java Virtual Machine)
Recap Static vs. Dynamic

```java
public class JustAdd {
    public int x;
    public int y;
    public int z;

    public static void main(String args[]) {
        x = 5;
        y = 10;
        z = x + y;
        System.out.println(z);
    }
}
```

Solution1: If non-OOP is intended

```java
public class JustAdd {
    public static int x;
    public static int y;
    public static int z;

    public static void main(String args[]) {
        x = 5;
        y = 10;
        z = x + y;
        System.out.println(z);
    }
}
```

Solution 2: If OOP is intended

```java
public class JustAdd {
    int x;
    int y;
    int z;
    //Method will executed by an object of Type JustAdd
    public int sumZ() {
        x = 5;
        y = 10;
        z = x + y;
        return z;
    }

    public static void main(String args[]) {
        JustAdd myAdd = new JustAdd(); //Main must create object first
        System.out.println(myAdd.sumZ());
    }
}
```

Summary of Static vs. Non Static

- Its better to write main method in a separate class so you do not get confused
- Major Points
  - Static variables and methods belong to class
    - To access them in another file we do filename.methodname() or filename.variablename()
  - Variables and methods not declared static automatic become OO
    - Then first we must create the object
Summary contd..

- Static methods besides main method can take in parameters of other object types
  - This allows static method to access to particular object of interest
  - Example:

    ```java
    public static void compareRadius(Circle c1, Circle c2) {
        if (c1.getRadius() >= c2.getRadius()) {
            System.out.println("Circle 1 is greater");
        }
    }
    ```