Data structures:

Introduction and getting started with Java

Dr. Sophea PRUM
sopheaprum@gmail.com

Static vs. dynamic typed languages

- Static-typed
  - Each variable is bound both to a **type** (declaration) and **object** (assignment)
  - The binding to an object is optional. If a variable is not bound to an object, it is said to be **null**.
  - Require variable declarration before being used

Static vs. dynamic typed languages

- Variable declaration in java

```
int     myNumber;
```

Type

Variable name

- Assignment

```
myNumber = 1;
```

- Variable declaration and initialization

```
int     myNumber = 1;
```

Static vs. dynamic typed languages

- Static-typed
  - Once a variable name has been bound to a type (that is, declared) it can be bound (via an assignment statement) only to objects of that type
  - It cannot ever be bound to an object of a different type.
  - An attempt to bind the name to an object of the wrong type will raise an error.

```
int myNumber=1;

String name="titi";

myNumber=name;
```

→ Error : incompatible type

Static vs. dynamic typed languages

- Dynamic-typed
  - Every variable is (unless it is null) bound only to an object at execution time by means of assignment statements
  - It is possible to bind a variable to objects of different types

myNumber = 1	myNumber is an interger
name = "titi"	name is a string
myNumber = name	myNumber is a string

Static vs. dynamic typed languages

Python is a dynamically-typed language. Java is a statically-typed language.

What is an algorithm ?

- **Code** is used to tell machine what to do. But before you code you need an **algorithm**.

- Algorithm : a **well designed** series of steps for solving a problem

- Code : interpretation of an algorithm into a computer programming language to order machines to execute

Algorithm # Code

Data type and data structure

- Data type : classification that specifies which type of value a variable has
  - Primitive data type : predefined by a computer programming language
  - Class type : template that used to create different objects.

Data type and data structure

- Primitive data type (can be different according to each programming language):

	Types	Size (bits)	Precision
Integer	byte	8	From +127 to -128
	char	16	All Unicode characters
	short	16	From +32,767 to -32,768
	int	32	From +2,147,483,647 to -2,147,483,648
	long	64	From +9,223,372,036,854,775,807 to -9,223,372,036,854,775,808
Floating-point	float	32	From 3.402,823,5 E+38 to 1.4 E-45
	double	64	From 1.797,693,134,862,315,7 E+308 to 4.9 E-324
Other	boolean	1	false, true
	void	--	--

Data type and data structure

- Data structure :
  - Logical way of organizing the data
  - Define the machanism to access data
  - Impact on your algorithm and system performance

  - Two types of data structures
    - Linear : Array, Stack, Queue, Linked list
    - Non-linear : Tree, Graph

Getting started with Java

Getting started with Java

- Java is an Object-Oriented Programming langage (OOP)
  - Run on Java Virtual Machine (JVM)
    - Platform independent (Windows, Linux, Mac OS)
  - Java Virtual Machine (JVM)
    - is an abstract computing machine that enables a computer to run a Java program
  - Java Runtime Environment (JRE)
    - is a software package that contains what is required to run a Java program including the implementation of JVM
  - Java Development Kit (JDK)
    - is a superset of a JRE and contains tools for Java programmers

Getting started with Java

- Java # Javascript
- Interface Development Environment (IDE)
  - **Netbeans**
  - Eclipse
  - Jcreator
  - Etc.

Java naming conventions

- Variable names can't start with a number. But numbers can be elsewhere

```
int     1stVariable; ⇒ Error

int     variable1; ⇒ OK
```

- Variable names can't be the same as Java keywords (*class, static, public, etc.*)
- Can not have spaces in your variable names
- Variable name start with lowercase character

Java naming conventions

- We've used the underscore character, but it's common practise to have the first word start with a lowercase letter and the second or subsequent words in uppercase:

  int myFirstNumber ;

  int mySecondNumber ;

- Variable names are case sensitive. So myfirstNumber and myFirstNumber are different variable names.
- Class name start with uppercase character and be a noun e.g. String, Color, Button, System, Thread etc.

Hello world!

web.xml | Start Page | HelloWorld.java

Source | History

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package helloworld;
7
8  /**
9   *
10  * @author sprum01
11  */
12 public class HelloWorld {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         System.out.println("Hello world!");
20         String name;
21
22     }
23 }
24
```

Package name

Class name

Main method

Java data type

- Java data types
  - Primitive types
    - Numeric types
      - Integer types
        - Integers
          - **byte**
          - **short**
          - **int**
          - **long**
        - Characters
          - **byte**
      - Floating point types
        - **float**
        - **double**
    - Boolean type
      - **boolean**
  - Reference types
    - Arrays
    - Interfaces
    - Classes
    - Enums

Control flow – If/else statement

```
if ( Statement ) {
    //do something
}
```

```
float mark= 50;
if ( mark < 50 ) {
    System.out.println("fail! :(");
}
else{
    System.out.println("Pass! :)");
}
```

Symbol	Meaning
==	Equal to
<	Smaller than
<=	Smaller than or equal
>	Greater than
>=	Greater than or equal
!	Not
!=	Different

Control flow – If/else statement

```
String result;
int age = 10;

if(age < 18){
    result = "User is under 18";
}
else if((age >=18)&&(age<=50)){
    result = "User is between 18 to 50";
}
else{
    result = "User is above 50";
}
System.out.println(result);
```

Symbol	Meaning
&&	And operator (short-circuit)
&	And operator, always evaluate both sides
||	Or operator (short-circuit)
|	Or operator, always evaluate both sides

Control flow – If/else statement

- Example: printing out month of the year according to a given month number using if/else

```
int monthNumber=1;
String monthName;
if(monthNumber == 1){
    monthName="January";
}
else if(monthNumber == 2){
    monthName="February";
}
```

Control flow – switch

```
switch ( variable_to_test ) {
    case value:
        code_here;
        break;
    case value:
        code_here;
        break;
    default:
        values_not_caught_above;
}
```

Control flow – switch

```
int month = 8;
String monthString;
switch (month) {
    case 1:  monthString = "January";
             break;
    case 2:  monthString = "February";
             break;
    case 3:  monthString = "March";
             break;
    case 4:  monthString = "April";
             break;
    case 5:  monthString = "May";
             break;
    case 6:  monthString = "June";
             break;
```

Control flow – for Loop

```
for ( start_value; end_value; increment_number ) {
    //Do something
}
```

```java
int startValue;
int endValue = 100;
int sum = 0;

for(startValue=1; startValue <= endValue; startValue++){
    sum = sum + startValue;
}
System.out.println("Sum = "+sum);
```

Control flow – while and do/while Loop

```
while ( condition ) {
    //Do something
}
```

```
do {
    //Do something
}while ( condition );
```

```
int maxValue = 10;
int initialValue = 1;

while(initialValue <= maxValue){
        initialValue++;
}
System.out.println(initialValue);
```

```
int maxValue = 10;
int initialValue = 1;

do{
    initialValue++;
}while(initialValue <= maxValue);
System.out.println(initialValue);
```

References

http://www.homeandlearn.co.uk/java/java.html
https://www3.ntu.edu.sg/home/ehchua/programming/java/J2a_BasicsExercises.html