

Core Java Khaing Swe Wynn (M.C.Sc)

Module 1

Khaing Swe Wynn, M.C.Sc United Label

What is Java?

Java is a portable across platform language designed based on the concept of object-oriented paradigm.

Sun was then acquired in 2009 by **Oracle**, which now owns and maintains Java.

It was created by James Gosling and Patrick Naughton, employees of **Sun Microsystems**, with the support of Bill Joy (co-founder of Sun 

Microsystems in 1982), officially presented on May 23, 1995 at SunWorld.

The peculiarity and central purpose of Java is that software written in this language must be very **easily portable** on several **operating systems** such as Unix, Windows, Mac OS or GNU / Linux, with little or no modifications.

Khaing Swe Wynn, M.C.Sc United Label

How to install and run Java on Windows

DownloadDownload Java SE Development Kits 8 ( jdk + jre)

•https://www.oracle.com/technetwork/java/javase/downloads/index.html

•JDK is for developing and testing the programs

•JRE is for running java programs but not for developing.

Install Install JDK first and JRE later.  

Check Check java folder including jdk and jre that has been installed in your  computer.

Type Type javac in your command line and check if it is working well or not.

Khaing Swe Wynn, M.C.Sc United Label

click

How to set up PATH in

environment variables

• Windows

1. Search System (Control Panel) 

2. Click the **Advanced system settings link.**

3. Click **Environment Variables**. In the section **System**

**Variables,** find the PATH environment variable and select it.

4. Click Edit. If the PATH environment variable does not exit, Click New.

5. In the **Edit System Variable** (or **New System Variable**)

window, specify the value of the PATH environment variable.

6. Click OK.

Khaing Swe Wynn, M.C.Sc United Label

Setting up Eclipse IDE

• Download the latest version of Eclipse from: http://www.eclipse.org/downloads.

Khaing Swe Wynn, M.C.Sc United Label

Creating an Eclipse Project

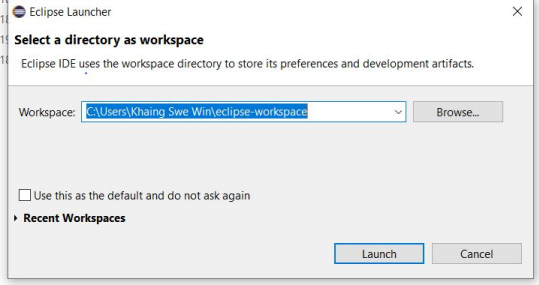
• Eclipse is a software development environment consisting of an integrated development environment (IDE) and an extensible plug-in system. 

1. Start Eclipse, using either a shortcut that should be available on a computer’s desktop or directly

executing “eclipse.exe” in an extracted directory. 2. Use default directory or browse to your own path in pop up dialog box

3. Optionally, you may switch to other workspaces via the menu “File → Switch Workspace → Others”

Khaing Swe Wynn, M.C.Sc United Label

Select workspace Khaing Swe Wynn, M.C.Sc United Label

Create Java Project

• Select ‘File → New → Java Project’ menu. 

• Enter a project name, e.g. MyFirstProject, and decide where the project should be saved or leave the default setting to use the current workspace. See Figure 2.

• Click ‘Next’ and then ‘Finish.’

Khaing Swe Wynn, M.C.Sc United Label

Build a java class file

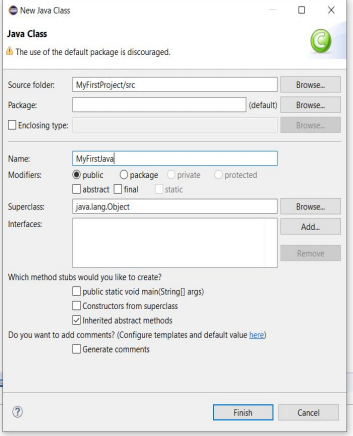
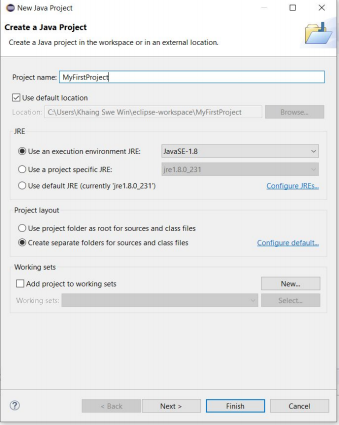
1. Select ‘File → New → Class’ menu 2. Enter a class name, e.g. MyFirstJava. 1. The main method is essentially the entry point to the application . 

2. Note that, in Python the main context is implicitly defined by lines having no

indent, but in Java it must be explicitly defined as the method.

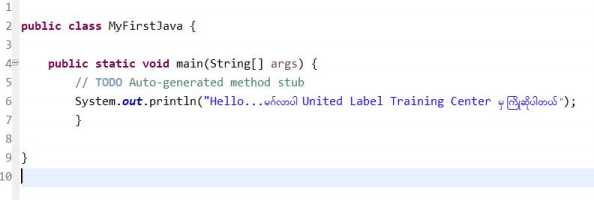
3. Click ‘Finish.’

Khaing Swe Wynn, M.C.Sc United Label

Khaing Swe Wynn, M.C.Sc United Label

Java Basics (Main Method) 

• Main method >> public static void main(String[] args) •Need “;” after one statement

• For comments -> use “//” or “/\*……………..\*/ “ Khaing Swe Wynn, M.C.Sc United Label

Java Basics >> Variable Declarations

• <variableType><variableName>; 

• <variableType><variableName> = <initialValue>;

• Examples :

• int num1;

• num1=100;

Khaing Swe Wynn, M.C.Sc United Label

Standard Data Types 

Khaing Swe Wynn, M.C.Sc United Label

Operators 

**Operations Types Operators Examples** Arithmetic Operations +, - , \*, /, % , ++, - -

Relational Operations ==, != , < , > , <= , >=

Logical Operations &&, || , !

Assignment Operations +=, -=, \*=, /=, %=,

Ternary Operations (condition) ? Value if true: value if false Bitwise Operations &, | , ~ , ^

Khaing Swe Wynn, M.C.Sc United Label

Operators Precedence

• Please check the following link:

• http://www.cs.bilkent.edu.tr/~guvenir/courses/CS101/op\_precedenc e.html

Khaing Swe Wynn, M.C.Sc United Label

If statement

• To make decisions, and to execute different parts of your program depending on a boolean true/false value.

• The if statement could be written in the following forms, where condition is either true or false. 

1. if(<condition>) { expression1 }

……………………………………..

xxxx……………………………………………………………… 2. if(<condition>) { expression 1}

else{ expression 2}

3. if(<condition1>) { expression 1}

else if(<condition2>){ expression 2}

else if(<conditionN>){ expression 3 }

Khaing Swe Wynn, M.C.Sc United Label

else{ expression 4}

Java Basics >> If Example Khaing Swe Wynn, M.C.Sc United Label

Java Basics >> While Loop 

• To repeat a block of statements while a condition is true. • The loop will be terminated if and only if the condition in the loop becomes false.

1.intialization;

while ( 2.Condition )

{

Statements

3.increasement or decreasement

}

Khaing Swe Wynn, M.C.Sc United Label

Java Basics >> while example

class WhileLoop {

public static void main(String[] args) {

int rowNum = 7;

int i = 0;

while (i < rowNum)

{

int j = 0;

while (j <= i)

{

System.out.print("#"); j++;

}

while (j < rowNum) {

System.out.print("\*"); j++;

}

System.out.println(); i++;

}

}



##\*\*\*\*\* ###\*\*\*\* ####\*\*\* #####\*\* ######\* #######

}

Khaing Swe Wynn, M.C.Sc United Label

Do- while loop

• The do-while statement is like the while statement, except that the associated block always gets executed at least once.

• Its syntax is as follows: 

1.intialization

Do

{

Statements;

2. increasement or decreasement;

} while (3. booleanExpression);

Khaing Swe Wynn, M.C.Sc United Label

Java Basics >> For Loop

• Like while loop, for loop consist of three operations:

(1) initialization of a variable,

(2) testing a condition, and 

(3) updating a value before the next iteration.

• The for loop groups these three common parts together into one statement, making it more readable and less error-prone than the while loop.

for(initialization; condition; increasement or decreasement)

{

Statements;

}

Khaing Swe Wynn, M.C.Sc United Label

Java Basics >> For Loop

class ForLoop {

public static void main(String[] args)

{

int rowNum = 7;

int j;

for (int i=0; i<rowNum; i++) {

for(j=0; j<=i; j++)

{

System.*out.print("#");*

}

for(int k=j;k<rowNum;k++)

{

System.*out.print("\*");*

}

System.*out.println();*

}

##\*\*\*\*\* ###\*\*\*\* ####\*\*\* #####\*\* ######\* #######

}

Khaing Swe Wynn, M.C.Sc United Label

}

Java Basics >> Enhanced For-loop

• Easier to write and make your code more readable. • To eliminate loop counter

• Syntax Form

For(data\_type item: collection or array)

{

}

\*\*\*The data type must be compatible with the array type. Khaing Swe Wynn, M.C.Sc United Label

Arrays

•A Java array is an ordered collection of primitives, object references, or other arrays.

• Each variable is referenced by array name and its index. Arrays may have one or more dimensions.

• Syntax

**type var\_name[ ]; ............... Step (1)**

**Var\_name=new type [ size ]; ............... Step (2)**

Example :

int days[];

days=new int[10]; = int days[]=new int[10]; Khaing Swe Wynn, M.C.Sc United Label

Initializing arrays

•int days[]={1,2,3,4,5,6,…...,10};

•int [][]multi=new int[2][];

•int [][]multi= { {1,2,3}, {4,5,6,7,8}};

For length of an array

• Length of an array => array.length

To copy an array

**• System.arraycopy**(sourceArray, source\_pos, destArray, dest\_pos,length) Khaing Swe Wynn, M.C.Sc United Label

To clone an array

•Array\_name.clone ();

Khaing Swe Wynn, M.C.Sc United Label

Using Java methods

Using Java methods

• The use of a method in Java is similar to that of a function in Python or C language. In general, a method in Java can be defined using the following syntax:

<modifier><returnedType><methodName> (<inputArguments>) 

**{ //Method body; }**

1. modifier: The modifier defines the access type of a method.

2. returnedType: A method may return a value. The returnedType is the data type of the value the method returns. In case that there is no returned value, returnedType is set as void. 3. methodName: This is the actual name of the method.

4. inputArguments: When a method is invoked, values may be passed to the arguments. 5. **Method Body:** The method body contains a series of Java statements defining what the method does.

Khaing Swe Wynn, M.C.Sc United Label

Method Example 

class Calculation{

static int max(int num1, int num2)

{

int result;

if (num1 > num2)

result = num1;

else result = num2;

return result;

}

public static void main(String[] args) { System.out.println("Max value is: " + max(5, 3));} }

Khaing Swe Wynn, M.C.Sc United Label

Reserved words or keywords

Khaing Swe Wynn, M.C.Sc United Label

Module 2

Khaing Swe Wynn, M.C.Sc United Label

Class, Class Components and dot operator

• To define a Java class, we use the class keyword follow with the name of the class:

class <className>{

...

}

• Class Variables

The variables inside a class definition are called class variables. class Circle{

double radius; float pi = 3.14f;

... }

Khaing Swe Wynn, M.C.Sc United Label

Class methods

• Class methods are nonstatic methods through the object instance. class Circle{

double radius;

float pi = 3.14f;

void printRadius()

{

System.out.println("Radius of this circle is "+radius); }

}

Khaing Swe Wynn, M.C.Sc United Label

Constructors

Constructors are methods with no returned type whose names are similar to the class name:

class Circle{

//constructor with no input parameters

Circle(){ ... }

//constructor with one input parameters

Circle(float radius){ ... }

//constructor with two input parameter

Circle(float radius, float pi){ ... }

}

Khaing Swe Wynn, M.C.Sc United Label

Objects and Instances

•An object is an instance of a class.

Two steps for an object of a class

1. **Initialization:** The new key word is used to create the object and followed by a call to a constructor.

2. **Assignment:** The object is assigned to the instance declared with an object type.

Circle circle = new Circle();

Khaing Swe Wynn, M.C.Sc United Label

Dot Operator

•Instance variables and methods of objects can be accessed via the dot(.) operators .

•Obj.variableName;

•Obj.methodName ();

Circle circle = new Circle();

circle.pi;

circle.printRadius();

Khaing Swe Wynn, M.C.Sc United Label

Using the this keyword

• To refer to the current object, Java provides the this keyword, which can be used in three ways:

1. as the object reference prefixing member variables, i.e. this.variableName,

2. as the object reference prefixing member methods, i.e. this.methodName(...), and

3. to invoke constructors, i.e. this(...).

Khaing Swe Wynn, M.C.Sc United Label

Using the static keyword

•When a variable is declared to be **static**, it can be accessed directly from within the class context, without the need to create an instance.

Khaing Swe Wynn, M.C.Sc United Label

Build a class called Circle

Khaing Swe Wynn, M.C.Sc United Label

Create a class called Account

Khaing Swe Wynn, M.C.Sc United Label

Create a class called AccountTester Khaing Swe Wynn, M.C.Sc United Label

Create a class called Maximum

Khaing Swe Wynn, M.C.Sc United Label

Visibility Modifiers

• private data fields and methods can only be accessed/modified from within the declaring class;

• protected modifier specifies that the member can only be accessed within its own package and, in addition, by a subclass of its class in another package.

• public data fields and methods can be accessed/modified by any class anywhere; and

• by default, data fields and methods are accessible from classes in the same package of the declaring class.

Khaing Swe Wynn, M.C.Sc United Label

Access Levels

Khaing Swe Wynn, M.C.Sc United Label

- , + , #



Khaing Swe Wynn, M.C.Sc United Label

Encapsulation

• The technique of making the variables in a class private and providing access to those variables via public methods.

Khaing Swe Wynn, M.C.Sc United Label

Implement a class in package p1 

Khaing Swe Wynn, M.C.Sc United Label

Implement Account in package p2 Khaing Swe Wynn, M.C.Sc United Label

Khaing Swe Wynn, M.C.Sc United Label

Inheritance

• The process of obtaining the data members and methods from one class to another class is known as **inheritance**.

• For designing such related classes in hierachy to avoid redundancy and promote code/software reuseability.

• To inherit a class, use the extends keyword

Khaing Swe Wynn, M.C.Sc United Label

Challenge for Inheritance

Khaing Swe Wynn, M.C.Sc United Label

Polymorphism

• Polymorphism is the concept of using a more general class variable to reference an object of a subclass.

• e.g., Vehicle v = new Bike().

Khaing Swe Wynn, M.C.Sc United Label

Abstract

• To create a superclass that defines only a generalized form that will be shared by all of its subclasses, leaving the detailed implementation to each subclass.

• Such a class determines the nature of the methods that the subclasses must implement, but does not provide an implementation of one or more of these methods.

• These classes are called abstract classes and are declared in Java using the abstract keyword.

• The abstract keyword is also used to declare an abstract method, • i.e. a method without an implementation.

Khaing Swe Wynn, M.C.Sc United Label

Interface

•Different from an abstract class, however, it cannot have data fields nor any concrete methods.

•An interface can be declared using the interface keyword.

• Methods declared in an interface can be inherited to a (abstract) class using the implements keyword.

Khaing Swe Wynn, M.C.Sc United Label

Challenge for Polymorphism Khaing Swe Wynn, M.C.Sc United Label

• Create a firm that includes 6 staff members who are 1. one executive whose pay is bonus plus payrate. 2. two employees whose payrate is similar to assigned payrate. 3. two volunteers who have not pay amount.

Khaing Swe Wynn, M.C.Sc United Label

Interface Challenge

Khaing Swe Wynn, M.C.Sc United Label

Khaing Swe Wynn, M.C.Sc United Label

Khaing Swe Wynn, M.C.Sc United Label

Hamburger Shop

•The purpose of the application is to help a fictitious company called Bills Burgers to manage their process of selling hamburgers.

•Our application will help Bill to select types of burgers, some of the additional items (additions) to be added to the burgers and pricing.

•We want to create **a base hamburger**, but also **two other types of hamburgers** that are popular ones in Bills store.

Khaing Swe Wynn, M.C.Sc United Label

•The basic hamburger should have the following items. 1. Bread roll type

2. meat and up to 4 additional additions (things like lettuce, tomato, carrot, etc) that the customer can select to be added to the burger.

•Each one of these items gets charged an additional price so you need some way to track how many items got added and to calculate the final price (base burger with all the additions). This burger has a base price and the additions are all separately priced (up to 4 additions, see above).

• Create a Hamburger class to deal with all the above. • **The constructor should only include the roll type, meat and price, can also include name of burger** or you can use a setter.

•• Also create two extra varieties of Hamburgers (subclasses) to cater for

a) Healthy burger (on a brown bread roll), plus two addition items that can be added.The healthy burger can have 6 items (Additions) in total.

• b) Deluxe hamburger - comes with chips and drinks as additions, but no extra additions are allowed.

•All 3 classes should have a method that can be called anytime to show the base price of the hamburger plus all additionals, each showing the addition name, and addition price, and a grand/final total for the burger (base price + all additions)

• For the two additional classes this may require you to be looking at the base class for pricing and then adding totals to final price.

Module 3

Khaing Swe Wynn, M.C.Sc United Label

Math

•Java provides a standard built-in Math class.

• As with the wrapper classes containing a series of **static** methods, the Math class is part of the **java.lang package** so, methods may be used without an explicit import statement (no instantiation)

Khaing Swe Wynn, M.C.Sc United Label

Khaing Swe Wynn, M.C.Sc United Label

Challenge for Math Function

Khaing Swe Wynn, M.C.Sc United Label

String Conversions

• Casting

• float v1 = 10.92f;

• int v2 = (int)v1;

\*\*\* Casting does not work as String and int are incompatible type for conversion either implicitly or explicitly. \*\*\* It requires an extra operation known as **parsing**.

• From String to int parsing,

•Integer.parseInt();

Khaing Swe Wynn, M.C.Sc United Label

Arrays and ArrayList

•Declaring an array and initializing array Int [] intArray=new int [10];

IntArray[0]=15;

IntArray[1]=30;

IntArray[2]=10;

Int [] intArray= {1,2,3,4,5,6,7,8,9,10};

Khaing Swe Wynn, M.C.Sc United Label

Challenge for Arrays

• Create a program using arrays that sorts a list of integers in ascending order. • Ascending order is lowest value to highest.

• In other words if the array had the values in it 106, 26, 81, 225, 15 your program should ultimately have an array with 15,26,81,106,225 in it.

• Set up the program so that the numbers to sort are read in from the keyboard. • Implement the following methods - getIntegers, printArray, and sortIntegers • getIntegers returns an array of entered integers from keyboard • printArray prints out the contents of the array

• and sortIntegers should sort the array and return a new array containing the sorted numbers

• you will have to figure out how to copy the array elements from the passed array into a new array and sort them and return the new sorted array.

Khaing Swe Wynn, M.C.Sc United Label

ArrayList

•ArrayList may be created with an initial size however when the size is exceeded, it is **automatically enlarged**.

• Likewise, when the objects are removed, the size of the ArrayList will be **shrunk automatically.**

•ArrayList<Integer> al1 = new ArrayList<Integer>();

•ArrayList<Double> al2 = new ArrayList<Double>();

•ArrayList<Student> al3=new ArrayList<Student>();

Khaing Swe Wynn, M.C.Sc United Label

Common Used Methods in ArrayList 

Khaing Swe Wynn, M.C.Sc United Label

Challenge for ArrayList

Khaing Swe Wynn, M.C.Sc United Label