Case Study: To define new data types:

1: Student Class

- Suppose we want to implement a class roster that contains both undergraduate and graduate students.
- Each student’s record will contain his or her name, three test scores, and the final course grade.
- The formula for determining the course grade is different for graduate students than for undergraduate students.

  Undergrads: pass if avg test score $\geq 70$
  Grads: pass if avg test score $\geq 80$

![Diagram of Student class and its subclasses Undergraduate and Graduate Student]
2: Patient Class

- Suppose we want to implement a patient class.
- Each patient record will contain his or her name, ID, and Room number.

```java
public class Patient {
    private String name;
    private String ID;
    private int RoomNo;

    public void setPatientDetails(String name, int RoomNo, String ID) {
        this.name = name;
        this.RoomNo = RoomNo;
        this.ID = ID;
    }

    public void displayPatientDetails() {
        // Draw everything
        System.out.println("Patient name is " + name);
        System.out.println("RoomNo is " + RoomNo);
        System.out.println("National Insurance ID is " + ID);
    }

    public String getName() {
        return name;
    }

    public String getID() {
        return ID;
    }

    public int getRoomNo() {
        return RoomNo;
    }
}
```

Multidimensional Arrays

- `int b[][] = {{1, 2}, {3, 4, 5}};`
- `int b[][];
  b = new int[3][4];`
- `int b[][];
  b = new int[2][]; // create 2 rows
  b[0] = new int[5]; // create 5 columns for row 0`
Problem Description:

How to sort an array and search an element inside it?

Solution:

Following example shows how to use sort () and binarySearch () method to accomplish the task. The user defined method printArray () is used to display the output:

```java
import java.util.Arrays;

public class MainClass {
    public static void main(String args[]) throws Exception {
        int array[] = { 2, 5, -2, 6, -3, 8, 0, -7, -9, 4 }
        Arrays.sort(array);
        printArray("Sorted array", array);
        int index = Arrays.binarySearch(array, 2);
        System.out.println("Found 2 @ "+ index);
    }
    private static void printArray(String message, int array[])
        System.out.println(message
            + ": [length: "+ array.length + "]");
            for (int i = 0; i < array.length; i++) {
                if (i != 0)
                    System.out.print(",");
            }
            System.out.println(array[i]);
        }
    System.out.println();
}
```

Result:

The above code sample will produce the following result.

```
Sorted array: [length: 10]
-9, -7, -3, -2, 0, 2, 4, 5, 6, 8
Found 2 @ 5
```

Lecturer: Maha Sabri Altememe
Problem Description:

How to search a word inside a string?

Solution:

This example shows how we can search a word within a String object using `indexOf()` method which returns a position index of a word within the string if found. Otherwise it returns -1.

```java
public class SearchStringEmp{
    public static void main(String[] args) {
        String strOrig = "Hello readers";
        int intIndex = strOrig.indexOf("Hello");
        if(intIndex == -1){
            System.out.println("Hello not found");
        }else{
            System.out.println("Found Hello at index "+ intIndex);
        }
    }
}
```

Result:

The above code sample will produce the following result.

```
Found Hello at index 0
```

-------------------------------------------------------------------------------------------------------

Problem Description:

How to use method overriding in Inheritance for subclasses?

Solution:

This example demonstrates the way of method overriding by subclasses with different number and type of parameters.

```java
public class Findareas{
    public static void main (String []args){
        Figure f= new Figure(10 , 10);
        Rectangle r= new Rectangle(9 , 5);
        Figure figref;
        figref=f;
        System.out.println("Area is ":+figref.area());
    }
}
```
Object Oriented Programming

Lecture (8)

```java
class Figure{
    double dim1;
    double dim2;
    Figure(double a , double b) {
        dim1=a;
        dim2=b;
    }
    double area() {
        System.out.println("Inside area for figure.");
        return(dim1*dim2);
    }
}

class Rectangle extends Figure {
    Rectangle(double a, double b) {
        super(a ,b);
    }
    double area() {
        System.out.println("Inside area for rectangle.");
        return(dim1*dim2);
    }
}
```

Result:
The above code sample will produce the following result.

```
Inside area for figure.
Area is :100.0
Inside area for rectangle.
Area is :45.0
```

```java
public static void main(String[] args)
{
    String firstString = "This is";
    String secondString = " a concatenated string.";
    String thirdString = firstString+secondString;
    System.out.println(thirdString);
    }  
```
Scanner scan = new Scanner(System.in);
for (i = 0; i < n; i++) {
    a[i] = scan.nextInt();
}

public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    int num[] = new int[10];
    int average = 0;
    int i = 0;
    int sum = 0;

    for (i = 0; i < num.length; i++) {
        System.out.println("enter a number");
        num[i] = in.nextInt();
        sum = sum + num[i];
    }
    average = sum / 10;
    System.out.println("Average=" + average);
}

public static void main(String[] args) {
    int i, n = 2;
    String a[] = new String[3];
    Scanner scan = new Scanner(System.in);
    for (i = 0; i <= n; i++) {
        System.out.print("enter a name:");
        a[i] = scan.next();
    }
    for (i = 0; i < a.length; i++) {
        System.out.println("print.. " + i + " name:" + a[i]);
    }
}