INPUT/OUPUT

Designing the User Interface

- A user interface is a program's input and output capabilities.
- An input operation is action that transfers data from the user to the computer's main memory.
- An output operation is action that transfers data from main memory to an output device.
- An command-line interface allows the user to type commands on a keyboard.
- A graphical user interface (GUI) uses graphical objects (windows, buttons) for I/O.

I) command-line interface

An command-line interface allows the user to type commands on a keyboard.

Basic Java program

Example 1

```
package firstproject;

public class FirstProject {

  public static void main(String[] args) {

      int first_number;
      first_number = 10;

      System.out.println("First number = " + first_number );

    }

}
```

Example 2

```
public static void main(String[] args) {

    int first_number, second_number, answer;

    first_number = 10;
    second_number = 20;
    answer = first_number + second_number;

    System.out.println("Addition Total = " + answer );
}
```

Accepting Input from a User

One of the strengths of Java is the huge libraries of code available to you. This is code that has been written to do specific jobs. All you need to do is to reference which library you want to use, and then call a method into action.

a) Scanner class

One really useful class that handles input from a user is called the **Scanner** class. The Scanner class can be found in the **java.util** library.

To use the Scanner class, you need to reference it in your code. This is done with the keyword import.
This tells java that you want to use a particular class in a particular library - the Scanner class, which is located in the **java.util** library.

The next thing you need to do is to create an object from the Scanner class. (A class is just a bunch of code. It doesn't do anything until you create a new object from it.)

Invoking a method on an object is to ask the object to perform a task. A Scanner object contains the following methods for reading an input:

- next(): reading a string. A string is delimited by spaces.
- nextByte(): reading an integer of the byte type.
- nextShort(): reading an integer of the short type.
- nextInt(): reading an integer of the int type.
- nextLong(): reading an integer of the long type.
- nextFloat(): reading a number of the float type.
- nextDouble(): reading a number of the double type.

To create a new Scanner object the code is this:

```
Scanner user_input = new Scanner( System.in );
```

Example

The program below requests user to enter first and family name using the Scanner class. The program then outputs the names on the screen.

```
package stringvars;

import java.util.Scanner;

public class StringVariables {

    public static void main(String[] args) {

        Scanner user_input = new Scanner(System.in);

        String first_name;
        System.out.print("Enter your first name: ");
        first_name = user_input.next();

        String family_name;
        System.out.print("Enter your family name: ");
        family_name = user_input.next();

        String full_name;
        full_name = first_name + " " + family_name;

        System.out.println("You are " + full_name);

    }
}
```

Example 2

```
//TestScanner.java
    import java.util.Scanner;             // import Scanner which is in java.util

    public class TestScanner {
     public static void main(String args[]) {
    // Create a Scanner
    Scanner scanner = new Scanner(System.in);

    // Prompt the user to enter an integer
     System.out.print("Enter an integer: ");
     int intValue = scanner.nextInt();
     System.out.println("You entered the integer " + intValue);

    // Prompt the user to enter a double value
    System.out.print("Enter a double value: ");
    double doubleValue = scanner.nextDouble();
    System.out.println("You entered the double value " + doubleValue);

     // Prompt the user to enter a string
    System.out.print("Enter a string without space: ");
    String string = scanner.next();
```

```
System.out.println("You entered the string " + string);
}
}
```

Example
Write a program that asks the user to input the length and width values during execution of the program and the area would be calculated based on the provided values.

```
import java.util.Scanner;
class AreaOfRectangle {
  public static void main (String[] args)
  {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the length of Rectangle:");
        double length = scanner.nextDouble();
        System.out.println("Enter the width of Rectangle:");
        double width = scanner.nextDouble();
        //Area = length*width;
        double area = length*width;
        System.out.println("Area of Rectangle is:"+area);
  }
}
```

NOTE ON SCANNER CLASS
When using Scanner, you may experience problem capturing data after you press the space bar. Normally if you use ***input. input.nex*t()** will stop reading while a white space is encountered. To resolve this, use **input.nextLine()** while taking a string as an input

```
import java.util.Scanner;
public class Example {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the full name of your course : ");
      String myString = scanner.nextLine();
      System.out.println("Entered string is : " + myString);
   }
}
```

b) Using BufferedReader for Input from the console

- A BufferedReader object performs buffered input. A buffer is memory where data is held until needed by the program.

Example
The program below connects the BufferedReader stream with the InputStreamReader stream for reading the line by line data from the keyboard

```
package bufferedprog;
import java.io.*;
public class BufferedProg {
   public static void main(String[] args)throws Exception {
                    InputStreamReader r=new InputStreamReader(System.in);
   BufferedReader br=new BufferedReader(r);
   System.out.println("Enter your name");
   String name=br.readLine();
   System.out.println("Welcome "+name);
   }

}
```