Lecture 5

- Covers
  - Algorithms (problem solving) using sequence, selection and repetition

Steps involved in solving problems on a computer

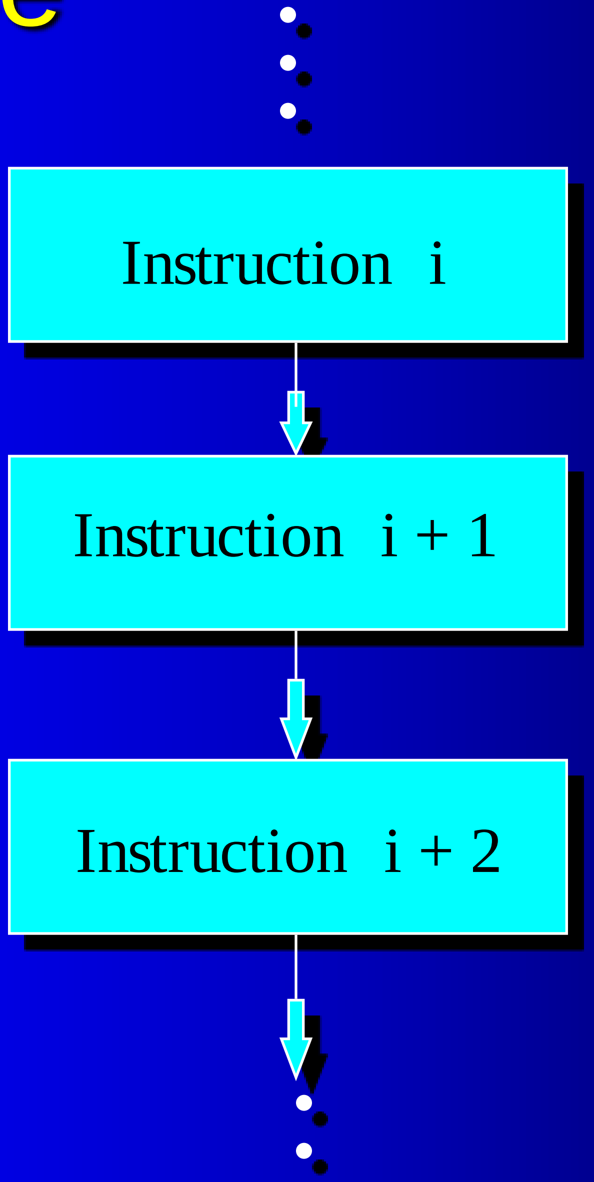
- Understand the problem
- Design a solution
- Implement (program) the solution
- Test the solution

► The three control structures

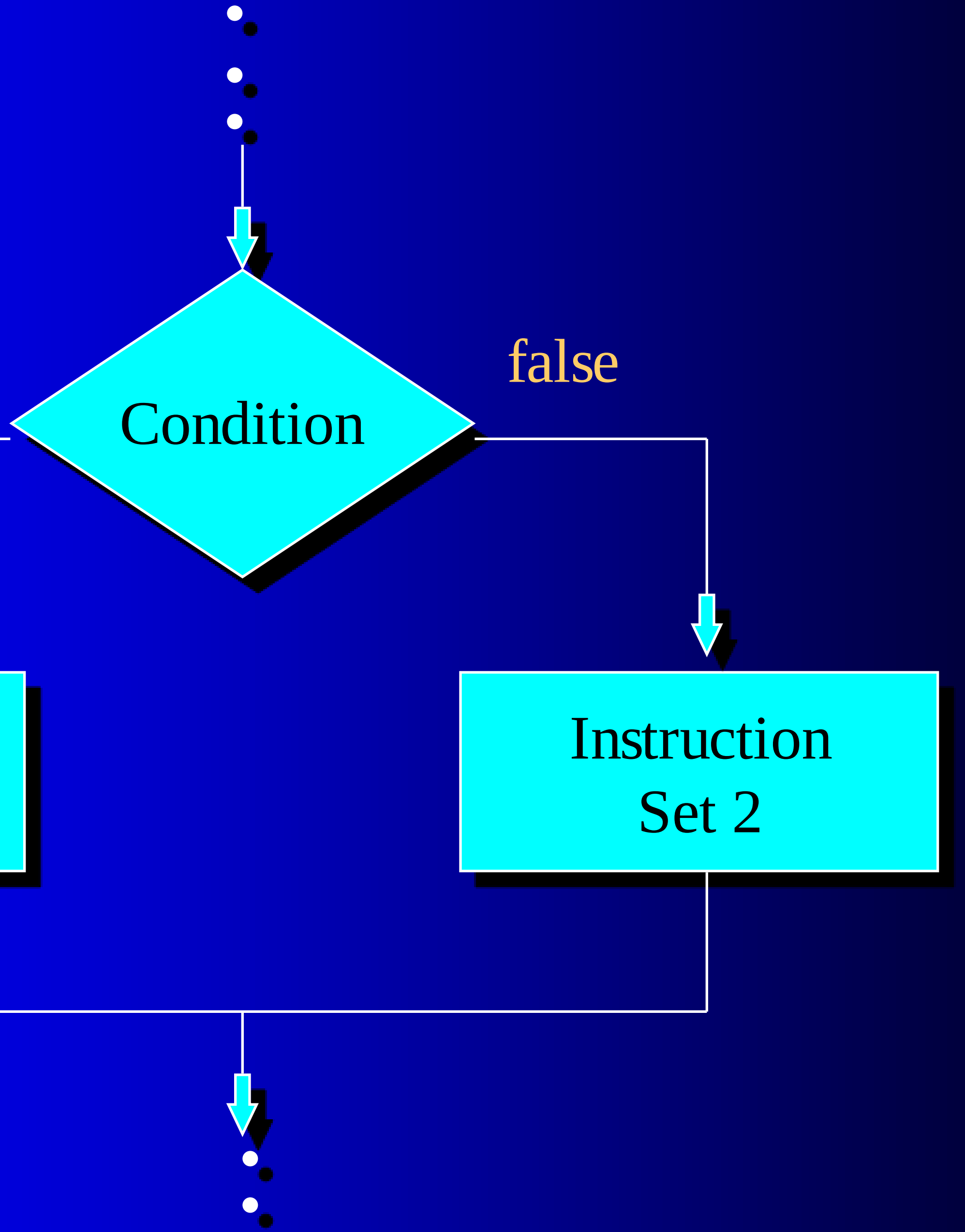
Control structures

- Sequence
  - Instructions executed in the order they are written
- Selection
  - Conditional execution of an instruction (or set of instructions)
- Repetition
  - Repeated execution of a set of instructions

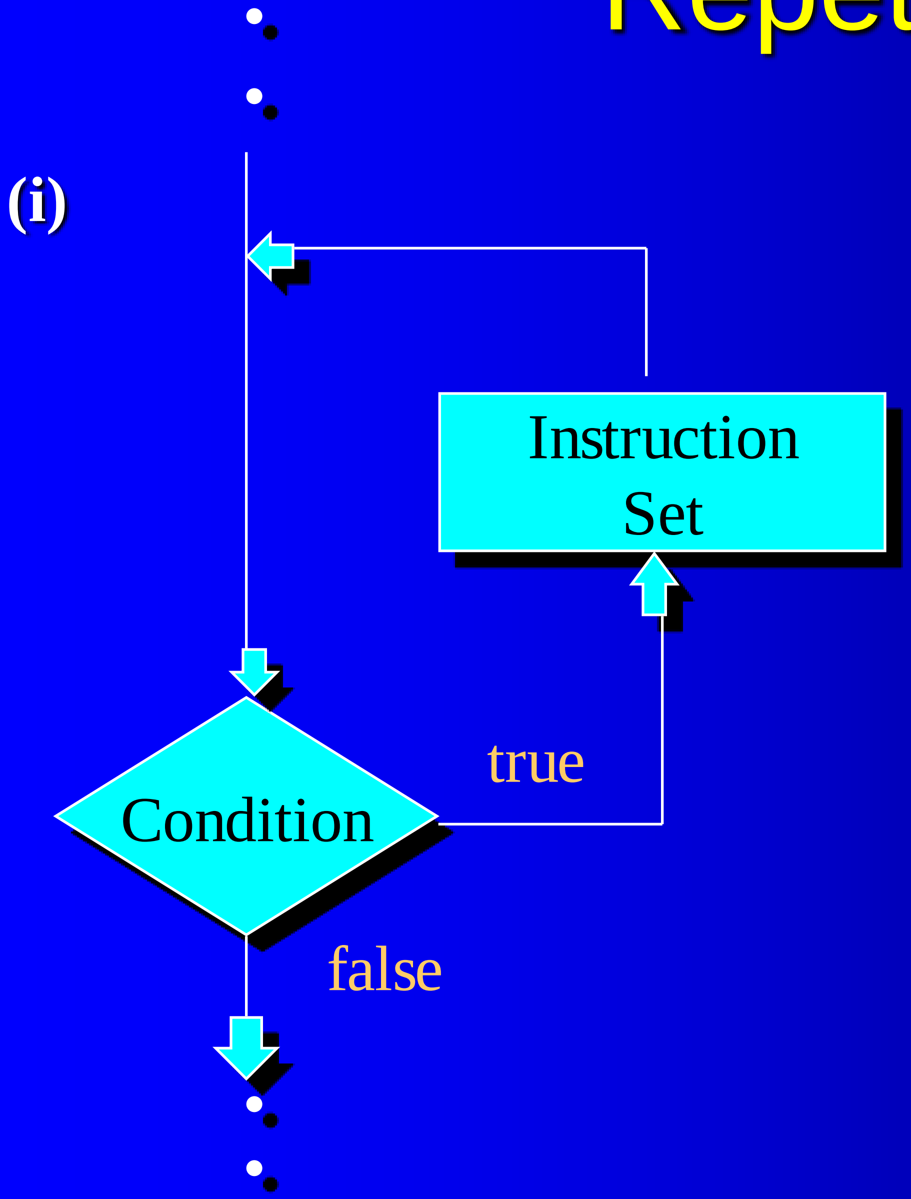
Sequence

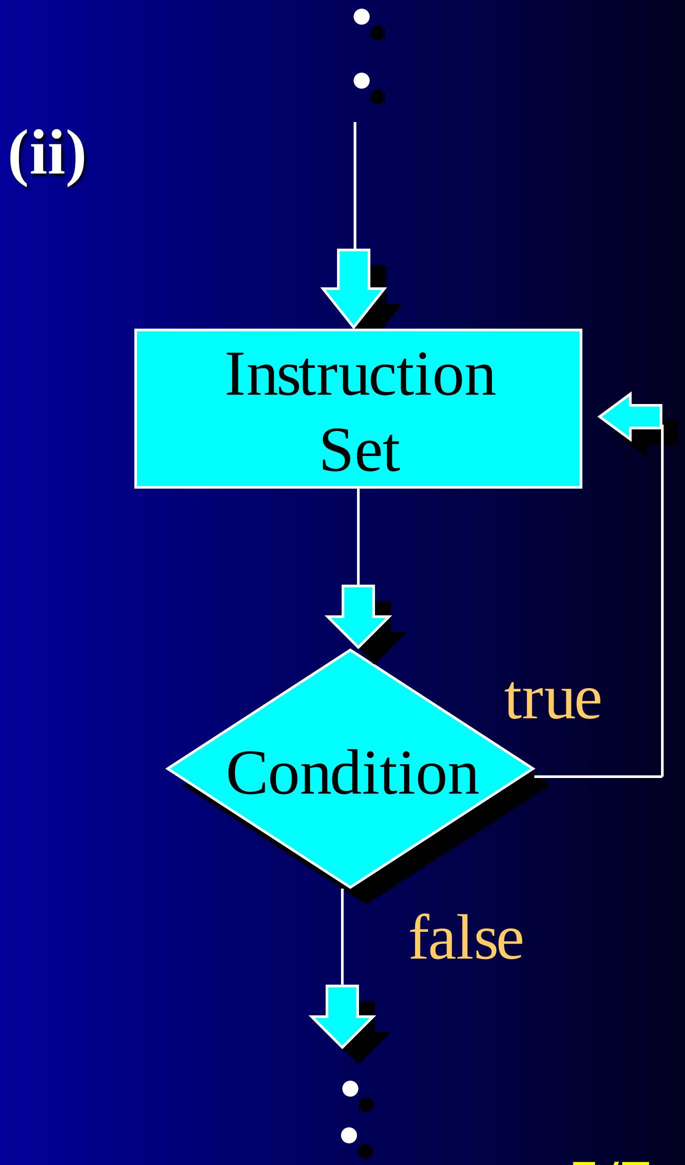


Selection



Repetition


(i)
Instruction Set
Condition
true
false


(ii)
Instruction Set
Condition
true
false

► Example 1

(Using sequence)

Sequence: average of three numbers

- Problem
  - Display the average of three numbers entered by the user
- Algorithm:

```
Get the first number
Get the second number
Get the third number
Calculate the average
Display the average
```

Sequence: average of three numbers

- In Java

```
import java.util.*;
public class Average
{
    public static void main(String[ ] args)
    {
        Scanner keyboard = new Scanner(System.in);
        System.out.println("Enter the three numbers: ");
        int n1 = keyboard.nextInt( );
        int n2 = keyboard.nextInt( );
        int n3 = keyboard.nextInt( );
        double average = (n1+n2+n3) / 3.0;
        System.out.println("The average is " + average);
    }
}
```

► Example 2

(Using selection)

Selection: maximum of two numbers

- Problem
  - Display the maximum of two numbers entered by the user
- Algorithm

```
Get the first number n1
Get the second number n2
IF n1 > n2 THEN
   Output n1
ELSE
   Output n2
ENDIF
```

Selection: maximum of two numbers

- In Java

```
import java.util.*;
public class Maximum
{
    public static void main(String[ ] args)
    {
        Scanner keyboard = new Scanner(System.in);
        System.out.println("Input  the two numbers:  ");
        int n1 = keyboard.nextInt( );
        int n2 = keyboard.nextInt( );
        System.out.print("Maximum  = ");
        if (n1 > n2)
        {
            System.out.println(n1);
        }
        else
        {
            System.out.println(n2);
        }
    }
}
```

► Example 3

(Using selection)

Determination of SubjectX pass

- Criteria for a pass
  - A student passes SubjectX if the student
    - Averages 50% or more on assignments and labs
    - Receives at least 40% in each exam
    - Gets 50% or more on the combined assignment/lab and exam marks where the assignments/labs contribute 30% and the exams contribute 70%

Determination of SubjectX pass

- Problem
  - Write a program to read in the assignment, lab and exam marks for a student and display “pass” or “fail” for each criterion, as well as the final mark
  - There will be 4 assignment marks, 2 lab marks and 2 exam marks

Determination of SubjectX pass

- Top level refinement
  - Express the problem in terms of major tasks and then solve each sub-task
- Solution

Are assignments and labs OK?

Are exams OK?

Is total mark OK?

Determination of SubjectX pass

- Refine sub-tasks
- Step 1: *Are assignments and labs OK?*
- Solution:

Determination of SubjectX pass

- Further refinement of step 1

```
Get assignment mark 1
Get assignment mark 2
Get assignment mark 3
Get assignment mark 4
Get lab mark 1
Get lab mark 2
average = (assign1 +assign2 +assign3 +assign4 +lab1 +lab2) /6
IF average >= 50 THEN
    Display “Passed assignment/lab hurdle!"
ELSE
    Display “Failed assignment/lab hurdle!"
ENDIF
```

Determination of SubjectX pass

- Refine subtasks
- Step 2: *Are exams OK?*
- Solution:

Determination of SubjectX pass

- Refine subtasks
- Step 3: *Is total mark OK?*
- Solution:

```
imprt java.util,*;
public class SubjectXPass
{
    public static void main(String[ ] args)
    {
        Scanner keyboard = new Scanner(System.in);
        boolean passedHurdle = true;
        System.out.println("Please enter 4 assignment marks and 2 lab marks: ");
        int assign1 = keyboard.nextInt( );
        int assign2 = keyboard.nextInt( );
        int assign3 = keyboard.nextInt( );
        int assign4 = keyboard.nextInt( );
        int lab1 = keyboard.nextInt( );
        int lab2 = keyboard.nextInt( );
        double pracAverage = (assign1 + assign2 + assign3 + assign4 + lab1 + lab2)
                                          / 6.0;
        if (pracAverage >= 50)
        {
            System.out.println("Passed assignment/lab hurdle!");
        }
        else
        {
            passedHurdle = false;
            System.out.println("Failed assignment/lab hurdle!");
```

Determination of SubjectX pass

```
System.out.println("Please enter 2 exam marks: ");

int exam1 = keyboard.nextInt( );
int exam2 = keyboard.nextInt( );

if ((exam1 >= 40) && (exam2 >= 40))
{
   System.out.println("Passed exam hurdle!");
}
else
{
   passedHurdle = false;
   System.out.println("Failed exam hurdle!");
}

double examAverage = (exam1 + exam2) / 2.0;
```

Determination of SubjectX pass

```
        double finalMark = 0.3 * pracAverage + 0.7 * examAverage;
        System.out.println("Final mark is " + finalMark + "%");

        if ((finalMark >= 50) && (passedHurdle == true))
        {
            System.out.println("Passed overall.");
        }
        else
        {
            System.out.println("Failed overall.");
        }
    }
}
```

► Example 4

(Using repetition)

SubjectX results

- Problem
  - Check the hurdle requirements and determine the final result for all students in the class
- Solution

SubjectX results

- Pseudocode solution

```
FUNCTION processStudentResult
    Get assignment/lab marks
    Check hurdle requirements
    Get exam marks
    Check hurdle requirements
    Compute final result
    Display final mark and pass or fail
ENDFUNCTION
```

SubjectX results

- To handle many students' results

```
WHILE (more students)
        processStudentResult
ENDWHILE
```

SubjectX results

- How do we know if there are any more students?

SubjectX results

- Pre-set number

```
int numberOfStudents = keyboard.nextInt( );
while (numberOfStudents > 0)
{
    // processStudentResult
    numberOfStudents = numberOfStudents - 1;
}
```

SubjectX results

- ‘Sentinel’ value
- Alter processing of a student’s result

```
int assign1 = keyboard.nextInt( );
while (assign1 >= 0)
{
   // processStudentResult
   assign1 = keyboard.nextInt( );
}
```

► Example 5

(exercise)

Class exercise: control structures

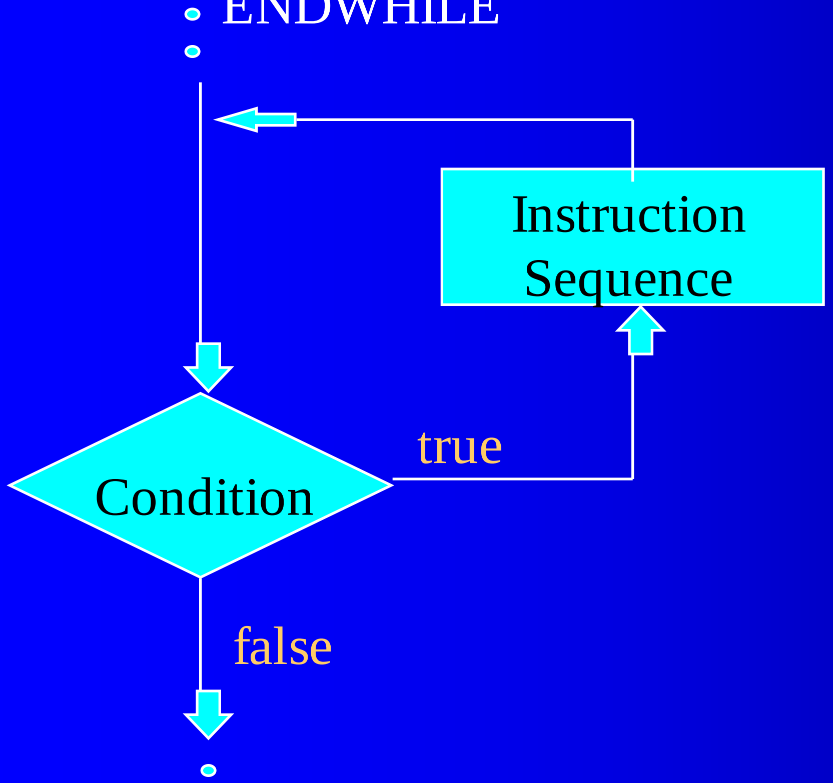
- Write pseudocode to solve the following problem
  - There is a (non-empty) line of people. Go to each person in the line and ask them their age. If they are older than 25, ask them to step forward.

A possible solution

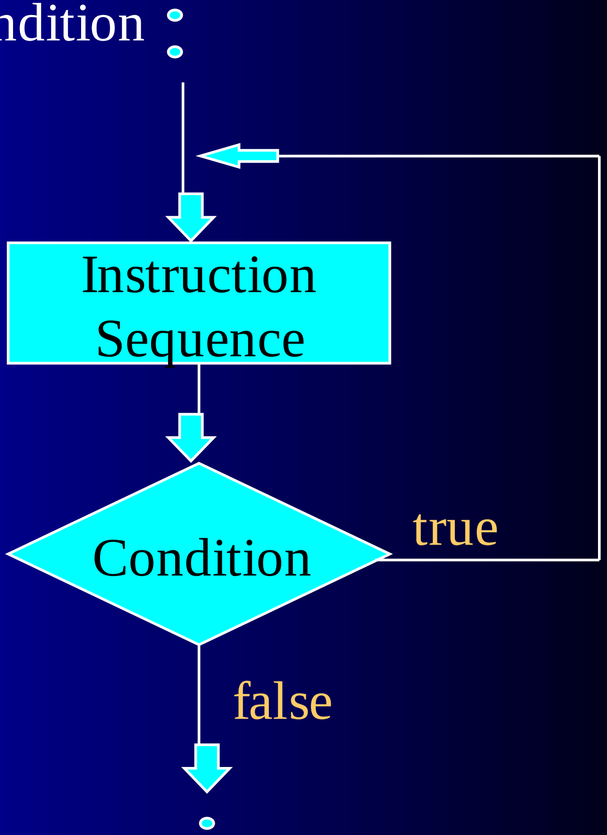
► WHILE loops vs. DO…WHILE loops

WHILE…ENDWHILE versus DO…WHILE loops

```
WHILE condition
    <instruction sequence>
ENDWHILE
```



```
DO
  <instruction sequence>
WHILE condition
```



DO...WHILE

- Problem
  - Write pseudocode to simulate crossing the road
- Basic actions
  - look left
  - look right
  - walk across
- Condition
  - road is busy

DO...WHILE

- Solution:

Class exercise

- Problem
  - Rewrite the solution to the "crossing the road" problem using the WHILE…ENDWHILE construct

Solution

WHILE…ENDWHILE versus DO...WHILE loops

- Using a WHILE…ENDWHILE loop to implement a DO…WHILE loop

```
DO
  <body>
WHILE condition
```

```
<body>
WHILE condition
    <body>
ENDWHILE
```

WHILE…ENDWHILE versus DO...WHILE loops

- Using a DO...WHILE loop and a selection control structure to implement a WHILE…ENDWHILE loop

```
WHILE condition
   <body>
ENDWHILE
```

```
IF condition THEN
   DO
      <body>
   WHILE condition
ENDIF
```

Next lecture

- Algorithms (problem solving) using functions (methods)
- Object-oriented analysis and design