Algoritma & Pemrograman Saintifik

Gatot F. Hertono, Ph.D

Departemen Matematika

SCMA601401

Algorithm

Definition

An algorithm is a finite set of precise instructions for performing a computation or for solving a problem.

Algorithms can be thought of as the recipe for taking the general solution for a class of problem and applying it to a specific instance of a problem covered by that class.

For instance, the class of problem might be to find the surface area of a sphere given its radius.

Algorithm:

```
TASK: Compute the surface area of a sphere
GET: radius
SET: area = 4pi*radius*radius
```

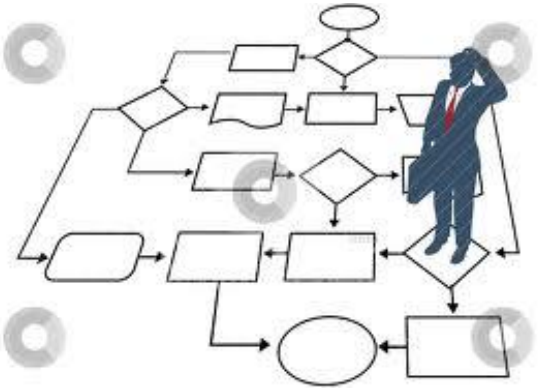
Source: http://www.dragonwins.com/courses/ECE1021/STATIC/LESSONS/AlgorithmRepresentation.htm#Algorithm%20Representation

Representing Algorithms

Since they are sets of instructions, they are generally presented in such a way that the step-by-step nature of how they should be followed is readily apparent.

The two most common representations are

- ✓ pseudocode and
- ✓ flowcharts.



Essential Elements of a Good Representation

- Show the logic of how the problem is solved - not how it is implemented.
- Readily reveal the flow of the algorithm (should involve DATA FLOW and CONTROL FLOW)
- Be expandable and collapsible.
- Lend itself to implementation of the algorithm.
- Implementation independence.

Source: http://www.dragonwins.com/courses/ECE1021/STATIC/LESSONS/AlgorithmRepresentation.htm#Algorithm%20Representation

Representing Algorithms

The properties of algorithms:

Complete

» For an algorithm to be complete, all of its actions must be exactly defined.

Unambiguous

» A set of instructions will be unambiguous if there is only one possible way of interpreting them.

Deterministic

» if the instructions are followed, it is certain that the desired result will always be achieved.

Finite

» the instructions must terminate after a limited number of steps.

Source: Paul Dunne, GMIT

Pseudocodes

Pseudocode is said as a program-like with syntax free (a mixture of natural language, mathematical notation and independent of any programming language).

ALGORITHM 1 Finding the Maximum Element in a Finite Sequence.

procedure $max(a_1, a_2, \ldots, a_n$: integers)
$max := a_1$
for $i := 2$ **to** n
    if $max < a_i$ **then** $max := a_i$
{max is the largest element}

Source: *Discrete Mathematics and Its Applications*, Kenneth H. Rosen

- There are very few commonly accepted standards for how pseudocode is written.
- This generally reflects the fact that it is used primarily as a rather short-term communication between members working on a specific project - the code itself and other documents are used for long-term archival purposes.
- pseudocode tends to be much more informal and a case of "whatever works".

Pseudocodes

Other examples:

Insertion Sort(A)

```
1  for j ← 2 to length(A)
2      key ← A(j)
3      i ← j-1
4      while i > 0 and A(i) > key

5          A(i+1) ← A(i)

6          i ← i – 1
       end while
7      A(i+1) ← key
```

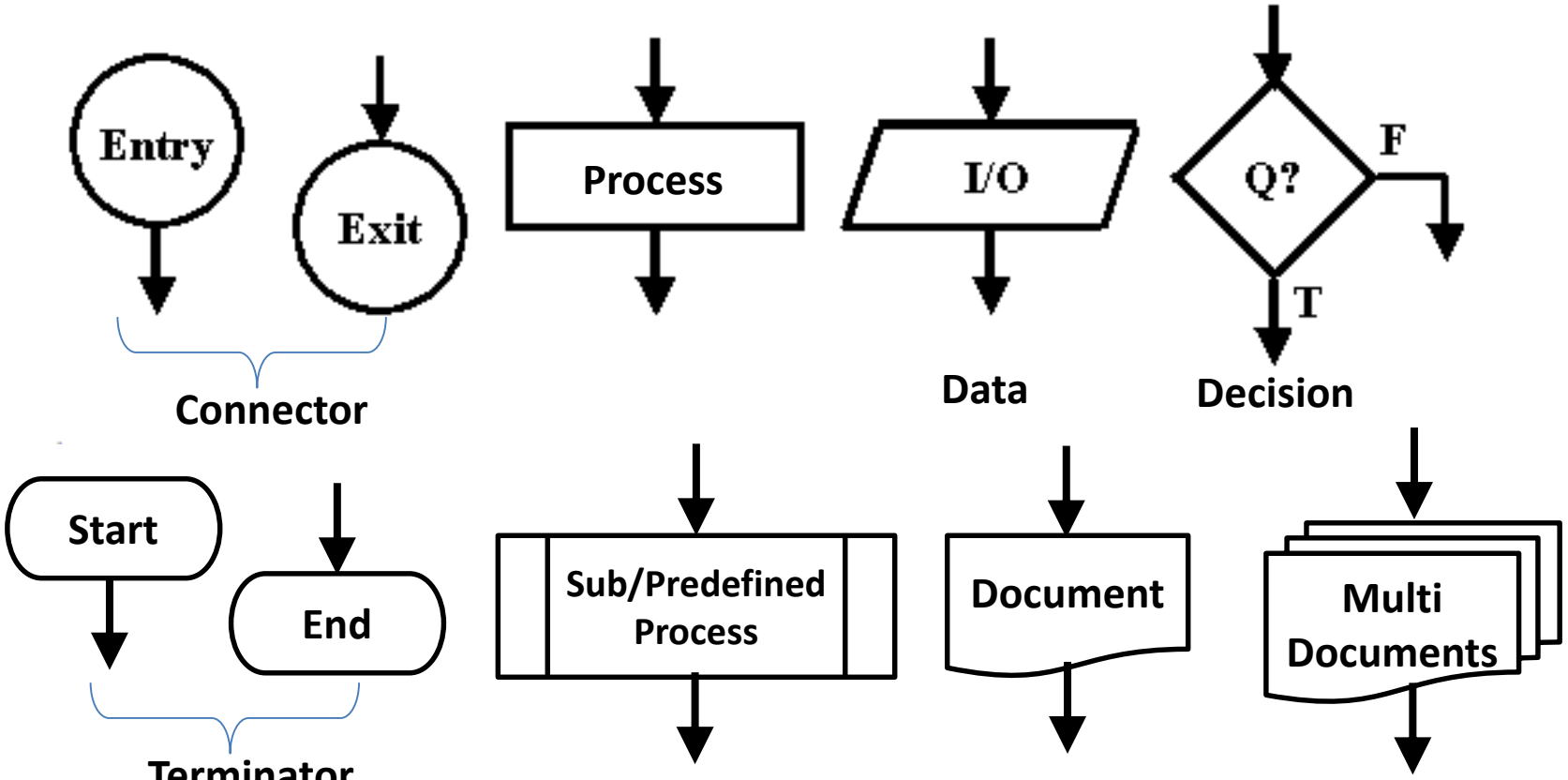
Recommended Format:
Assignment (:=, ←)
Input/Output (READ/INPUT, PRINT/WRITE)
Flow Control:
- Conditions (IF … THEN … ELSE … , CASE/SELECT …)
- Loop (FOR … TO … , WHILE … , REPEAT … UNTIL …)

Flowcharts

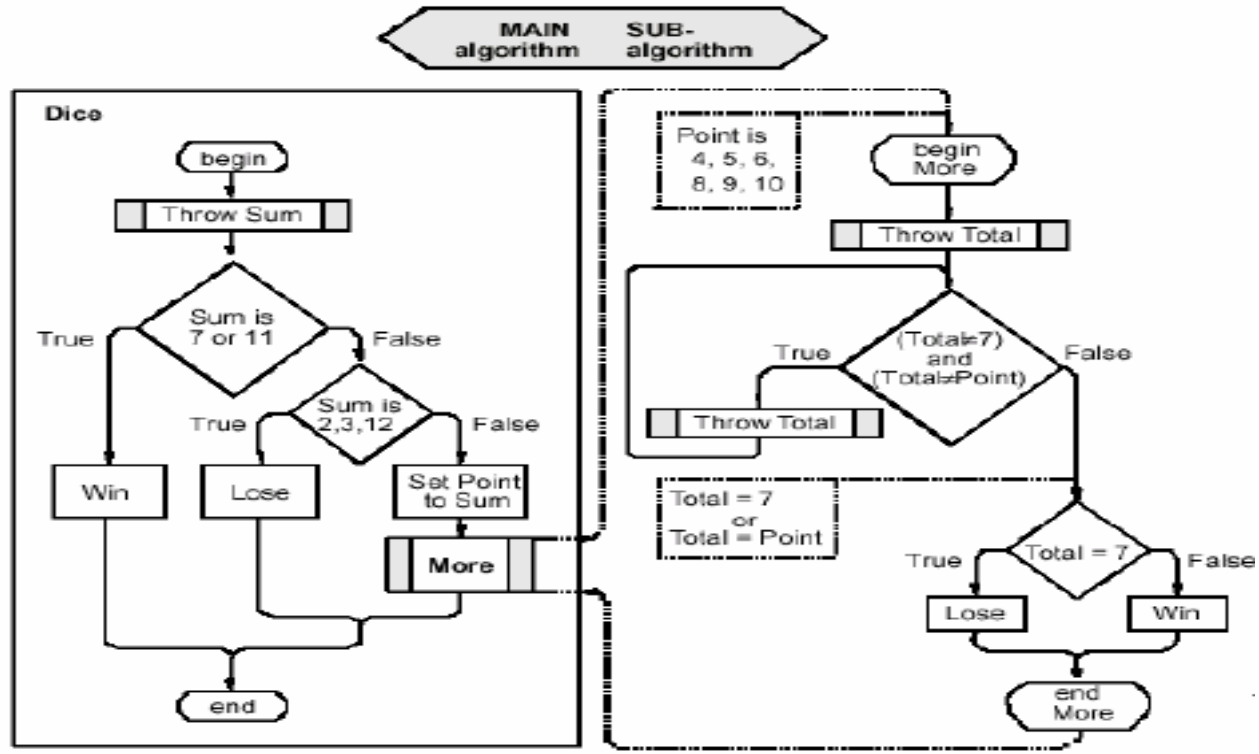
Flowcharts are a graphical means of representing an algorithm. Flowcharts tend to be the preferred means because they **convey structure** much more effectively.

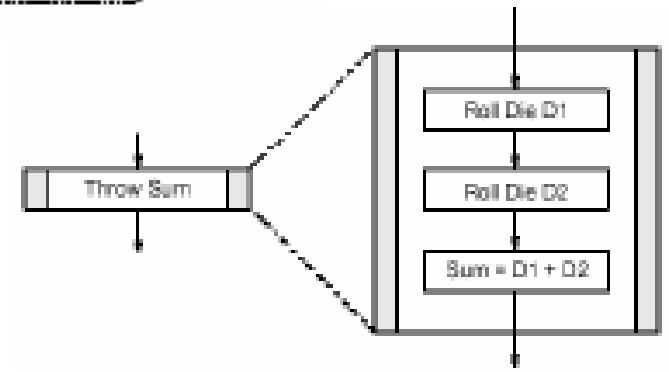
Basic Flowchart Shapes

The shapes we will use are the circle, oval, the rectangle, the parallelogram, the diamond, and the arrows that interconnect them.



Flowcharts: an example





Tugas Minggu 2

Buatlah algoritma yang direpresentasikan dalam pseudo code dan flowchart untuk 2 problem yang telah diberikan pada minggu 1 (n! dan bilangan Fibonacci).