Algorithm and Flowchart
Programming Methodology

Problem solving

• Problem statement and analysis
• Develop a high-level algorithm
• Detail out a low-level algorithm

Coding

• Choose a programming language
• Code the program using the selected algorithm
• Test the program and correct the errors
Algorithm

Definition – Solution to a computer programming problem.

Algorithm can be written in 2 different ways

- **Pseudo-code** – English-like steps that describes the solution
- **Flowcharts** – Picture with specific blocks detailing out the logical flow of the solution
Flowchart Building Blocks

CONTROL FLOW
TERMINAL POINT - Start / End
PROCESS - Initializing, Calculation ...
INPUT / OUTPUT - Keyboard, Display ...
DECISION
CONNECTOR - used for big diagram across pages
PRINTOUT
STORAGE - Read or Write from CDs, Disks, Tapes
SUB-ROUTINE
**Example 1**

**Problem Statement**
Watch a movie at home

**Algorithm**
1. Switch on the TV and UBC sets
2. Change to the required movie channel
3. Sit down and watch the movie
Example 2

Problem Statement
Withdraw cash from ATM

Algorithm
1. Go to the ATM
2. Insert your card into the machine
3. Press in your code
4. Choose “Withdraw” and enter Amount required
5. Take the cash, slip and card.
Example 3

Problem Statement
Calculate the interest of a bank deposit. You are to read the amount, years and interest rate from the keyboard and print the interest amount.

Algorithm
1. Read Amount
2. Read Years
3. Read Rate
4. Set Interest as Amount * Rate * Years / 100
5. Print Interest
Example 3 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount = 5000</td>
<td>Interest = 200</td>
</tr>
<tr>
<td>Years = 2</td>
<td></td>
</tr>
<tr>
<td>Rate = 2</td>
<td></td>
</tr>
<tr>
<td>Amount = 1000</td>
<td>Interest = 37.50</td>
</tr>
<tr>
<td>Years = 1.5</td>
<td></td>
</tr>
<tr>
<td>Rate = 2.5</td>
<td></td>
</tr>
</tbody>
</table>
Example 4

**Problem Statement**
Print what to do when driving to a traffic signal

**Algorithm**
1. Read traffic signal
2. If signal is GREEN then
   Set Action as GO
   Else
   Set Action as STOP
3. Print Action
Example 4 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal = GREEN</td>
<td>Action = GO</td>
</tr>
<tr>
<td>Signal = RED</td>
<td>Action = STOP</td>
</tr>
<tr>
<td>Signal = YELLOW</td>
<td>Action = STOP</td>
</tr>
<tr>
<td>Check what happens if Signal = BLUE</td>
<td>Action =</td>
</tr>
</tbody>
</table>
Example 5

Problem Statement
Read a number from the keyboard. Check and output if a given number N is ODD or EVEN

Algorithm
1. Read N
2. Set Remainder as N modulo 2
3. If Remainder is equal to 0 then
   Set Answer as EVEN
   Else
   Set Answer as ODD
4. Print Answer
## Example 5 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>N = 5</td>
<td>Answer = ODD</td>
</tr>
<tr>
<td>N = 8</td>
<td>Answer = EVEN</td>
</tr>
<tr>
<td>N = 0</td>
<td>Answer = EVEN</td>
</tr>
<tr>
<td>N = -1</td>
<td>Answer = ODD</td>
</tr>
</tbody>
</table>
Example 6

Problem Statement
Print Title for a person (Either Mr. or Miss. or Mrs.). You are to read the gender (and status if needed).

Algorithm
1. Read Gender
2. If Gender is MALE then
   Title is Mr.
   Else
   Read Status
   If Status is MARRIED then
      Title is Mrs.
   Else
      Title is Miss.
3. Print Title
## Example 6 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender = Male</td>
<td>Title = Mr.</td>
</tr>
<tr>
<td>Gender = Female</td>
<td>Title = Mrs.</td>
</tr>
<tr>
<td>Status = Married</td>
<td></td>
</tr>
<tr>
<td>Gender = Female</td>
<td>Title = Miss.</td>
</tr>
<tr>
<td>Status = Single</td>
<td></td>
</tr>
<tr>
<td>Check what happens if</td>
<td>Title =</td>
</tr>
<tr>
<td>Gender = Boy</td>
<td></td>
</tr>
<tr>
<td>Status = Intelligent</td>
<td></td>
</tr>
</tbody>
</table>
Example 7

Problem Statement
Print 1 to 20

Algorithm
1. Initialize X as 0
2. Increment X by 1
3. Print X
4. If X is less than 20 then go back to Step 2

Start
Initialize X <- 0
Increment X by 1
Print X
Yes
X < 20
No
End
Example 8

Problem Statement
Given computer time is stored in 24 hours format, you are to print the time in AM/PM format.

Algorithm
1. Retrieve computer time
2. Extract Hours and Minutes
3. If Hours is equal to 0 then
   Print 12
   Else
      If Hours is between 1 and 12 then
         Print Hours
      Else
         Print Hours – 12
4. Print ‘:’
5. Print Minutes
6. If Hours is less than 12 then
   Print AM
   Else
   Print PM

204112 Structured Programming
### Example 8 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer time = 8:30</td>
<td>Printed time – 8:30 AM</td>
</tr>
<tr>
<td>Computer time = 20:30</td>
<td>Printed time – 8:30 PM</td>
</tr>
<tr>
<td>Computer time = 0:15</td>
<td>Printed time – 12:15 AM</td>
</tr>
<tr>
<td>Computer time = 12:15</td>
<td>Printed time – 12:15 PM</td>
</tr>
</tbody>
</table>
Example 9

Problem Statement
Read the Month (and Year, if needed) and print the number of days in that month

Algorithm
1. Read MONTH
2. If MONTH is equal to 2 then
   Read YEAR
   If YEAR is a leap year then
     Set DAYS as 29
   Else
     Set DAYS as 28
   Else
     If MONTH is either 4 or 6 or 9 or 11 then
       Set DAYS as 30
     Else
       Set DAYS as 31
3. Print DAYS
### Example 9 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Month = 2, Year = 2004</td>
<td>Days = 29</td>
</tr>
<tr>
<td>Month = 2, Year = 2005</td>
<td>Days = 28</td>
</tr>
<tr>
<td>Month = 10</td>
<td>Days = 31</td>
</tr>
<tr>
<td>Month = 4</td>
<td>Days = 30</td>
</tr>
<tr>
<td>Check what happens if Month = -1</td>
<td>Days =</td>
</tr>
</tbody>
</table>
Example 10

Problem Statement
Prepare sandwiches

High-level Algorithm
1. Go to the nearest supermarket
2. Pick the groceries you need
3. Pay at the cashier
4. Bring the groceries home
5. Prepare the sandwiches

Low-level Algorithm
1.1 Take the car keys and wallet from the counter
1.2 Drive the car to the supermarket
1.3 Park the car
1.4 Take the lift to the supermarket floor
2.1 Take an empty cart and walk around the floor
2.2 Put the needed groceries into the cart
2.3 Take the cart to the cashier
3.1 Give the credit card to the cashier
3.2 Sign on the credit card slip
4.1 Take the cart with the plastic bags to the car
4.2 Put the plastic bags to the car
4.3 Drive the car home
4.4 Remove the plastic bags from the car
5.1 Cut the bread into half
5.2 Prepare the bacon and salad
5.3 Put the ingredients between 2 slices of bread
Example 11

**Problem Statement**
Make an urgent call to your friend from the airport

**High-level Algorithm**
1. Go to a public booth
2. Dial your friend’s number
3. Give the message to your friend

**Low-level Algorithm**
1.1 Walk to the next phone booth
1.2 If phone booth is not working, then repeat from step 1.1
2.1 Retrieve the number from pocket diary
2.2 Put some coins into the slot.
2.3 Dial the number
2.4 If the line is busy, hang up, then take back the coins and repeat from step 2.2
3.1 If your friend can come to the phone, then talk to your friend.
3.2 If your friend cannot come to the phone, then leave a message for your friend.
3.2 Hang up the phone.
3.4 Retrieve any coins not used.
Example 11

Start

Walk to next phone booth

Phone working?

YES

Retrieve Number from Pocket Diary

Put some coins into the slot
Dial the number

Line is busy?

YES

Hang up the phone
Retrieves the unused coins

NO

YES

Friend found?

YES

Talk to friend

NO

Leave message

End
Example 12

Problem Statement
Automatically return change for a purchase of N baht when given a 20 baht note. Check that N is between 1 and 20.

High-level Algorithm
1. Read and Validate N
2. Calculate Change
3. Decide how many 10 baht coins, 5 baht coins and 1 baht coins to return

Low-level Algorithm
1.1 Read N
1.2 If NOT (1 <= N <= 20) then
   Print Error Message
   Go back to Step 1.1
2.1 Initialize CHANGE as 20
2.2 Deduct N from CHANGE
3.1 If CHANGE is less than 10 then
   Number of 10 baht coin is 0.
   Else
   Number of 10 baht coin is 1.
   Deduct 10 from CHANGE
3.2 If CHANGE is less than 5 then
   Number of 5 baht coin is 0.
   Else
   Number of 5 baht coin is 1.
   Deduct 5 from CHANGE
3.3 Number of 1 baht coin is CHANGE

What happens if customer can pay by any kinds of banknotes: 1000, 500, 100, 20, and 10. and any kinds of coins: 10, 5, 2, and 1. That means N is not be fixed.
Example 12

Check what happens if $N = 20$
Check what happens if $N = 0$
Check what happens if $N = 21$
## Example 12 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>N = 17</td>
<td>Number of 10 B coin – 0</td>
</tr>
<tr>
<td></td>
<td>Number of 5 B coin – 0</td>
</tr>
<tr>
<td></td>
<td>Number of 1 B coin – 3</td>
</tr>
<tr>
<td>N = 6</td>
<td>Number of 10 B coin – 1</td>
</tr>
<tr>
<td></td>
<td>Number of 5 B coin – 0</td>
</tr>
<tr>
<td></td>
<td>Number of 1 B coin – 4</td>
</tr>
<tr>
<td>N = 13</td>
<td>Number of 10 B coin – 0</td>
</tr>
<tr>
<td></td>
<td>Number of 5 B coin – 1</td>
</tr>
<tr>
<td></td>
<td>Number of 1 B coin – 2</td>
</tr>
<tr>
<td>N = 20</td>
<td></td>
</tr>
<tr>
<td>N = 0</td>
<td></td>
</tr>
<tr>
<td>N = 21</td>
<td></td>
</tr>
</tbody>
</table>
Example 13

Problem Statement
Find the average of a given list of numbers

High-level Algorithm
1. Find the SUM of the given numbers
2. Find the COUNT of the given numbers
3. AVERAGE is SUM \div \text{COUNT}

Low-level Algorithm
1. Initialize SUM as 0 and COUNT as 0
2. If there are no more numbers remaining to be processed, then go to step 7.
3. Set ITEM as next number in the list
4. Add ITEM to SUM
5. Increment COUNT by 1
6. Go back to step 2
7. If COUNT is equal to 0, then
   \hspace{1cm} AVERAGE is “undefined”
Else
   \hspace{1cm} AVERAGE is SUM \div \text{COUNT}
**Example 13**

Start

Set SUM <- 0  
Set COUNT <- 0

NO

Any more unprocessed numbers?

YES

Set ITEM <- Next number in list  
Set SUM <- SUM + ITEM  
Set COUNT <- COUNT + 1

YES

COUNT = 0?

NO

Set AVERAGE = SUM / COUNT

YES

Output "Undefined" Error

NO

Output AVERAGE

End
### Example 13 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>List = 20, 2, 5, -3</td>
<td>Average = 6</td>
</tr>
<tr>
<td>List = 2, 5, -3, -8, -1</td>
<td>Average = -1</td>
</tr>
<tr>
<td>List = 2, 7, 5, 3, 6</td>
<td>Average = 4.60</td>
</tr>
<tr>
<td>List = 4</td>
<td>Average = 4</td>
</tr>
<tr>
<td>List =</td>
<td>Average = “undefined”</td>
</tr>
</tbody>
</table>
Example 14

Problem Statement
Given a 2-D polygon with N sides (and N vertices). Find the smallest rectangular box required to cover the polygon completely

Algorithm
1. Initialize MINX, MINY, MAXX, MAXY using the 1st Vertex
2.Retrieve the next unevaluated vertex (X, Y)
3. If X < MINX, then set MINX as X
4. If X > MAXX, then set MAXX as X
5. If Y < MINY, then set MINY as Y
6. If Y > MAXY, then set MAXY as Y
7. If all vertices have not been evaluated then go back to step 2
8. Set LENGTH as MAXX – MINX
9. Set HEIGHT as MAXY – MINY
Example 14

Start

Get the 1st Vertex (X,Y)

Set MINX <- X
Set MAXX <- X
Set MINY <- Y
Set MAXY <- Y

Get the next Vertex (X,Y)

X < MINX
   YES => Set MINX <- X
   NO

X > MAXX
   YES => Set MAXX <- X
   NO

Y < MINY
   YES => Set MINY <- Y
   NO

Y > MAXY
   YES => Set MAXY <- Y
   NO

Set LENGTH <- MAXX - MINX
Set HEIGHT <- MAXY - MINY

End

Any unevaluated vertex left?

NO

AB
B
Any unevaluated vertex left?

YES

A

NO

End
# Example 14 – Input/Output Samples

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
</table>
| 4 sides (2,2) (5,3) (3,5) (6,2) | Length = 4  
|                            | Height = 3 |
| 3 sides (1,2) (5,3) (8, -2) | Length = 7  
|                            | Height = 5 |
| 5 sides (2,5) (7,1) (3,2) (-3, -5) (4,1) | Length = 10  
|                            | Height = 10 |