POLITEKNIK MALAYSIA
KUALA TERENGGANU

KEMENTERIAN PENGAJIAN TINGGI

KOLEJ KOMUNITI MALAYSIA

QUICK LEARNER : PROBLEM SOLVING & PROGRAM DESIGN

(2ND EDITION)

Penulis :
YUSNITA HALIM
NAPISAH HARUN
ROSMAYATI ISMAIL

TEKNOLOGI & PERKOMPUTERAN

Quick Learner :
Problem Solving & Program Design (2nd Edition)

First Published 2021
Published 2021 (electronic)

Politeknik Kuala Terengganu

eISBN 978-967-2240-20-4

Author :

Yusnita Halim
Napisah Harun
Rosmayati Ismail

Published by :

Politeknik Kuala Terengganu,
20200 Jalan Sultan Ismail,
Kuala Terengganu, Terengganu.

Perpustakaan Negara Malaysia Cataloguing-in-Publication Data
Yusnita Halim, 1979-
QUICK LEARNER : PROBLEM SOLVING & PROGRAM DESIGN / YUSNITA HALIM, NAPISAH HARUN, ROSMAYATI ISMAIL. - 2ND EDITION.
Mode of access: Internet
eISBN 978-967-2240-20-4
1. Programming languages (Electronic computers).
2. Problem solving--Data processing.
3. Programming (Electronic computers).
4. Government publications--Malaysia.
5. Electronic books.
I. Napisah Harun, 1984-. II. Rosmayati Ismail, 1976-.
III. Title.
005.133

ACKNOWLEDGEMENTS

Praise Allah the Almighty for the knowledge and strength that He has granted upon us in writing this book. We would like to express our deepest appreciation to our team members for their unparalleled support towards the completion of this book.

To our families, the completion of this book would not have been possible without their encouragement, care and support.

Our appreciation also goes out for our friend and colleagues. Thank you for your advice, support, and constructive critics, etc. We are really grateful for your support and friendship.

To our former students, thank you for your comments and feedback during the process for producing this book. For our students and others who will be using this book, we hope that this book will provide a better understanding in problem solving and program design.

Finally, to everyone that have helped us in the production of this book, please accept our deepest thanks. May Allah bless you.

Yusnita Halim
Napisah Harun
Rosmayati Ismail

PREFACE

This book is written specifically for the use of students who are studying Problem Solving and Program Design course in the form of a simpler infographic. The syllabus is written for polytechnic and foundation level for problem solving using C++ language, however, it can also be a useful guide for undergraduates.
For each topic in this book, there are examples of activities and solutions provided. In addition, exercise questions were provided for each topics to enhance student's understanding. The book has been organized by chapter.

Chapter 1 : Introduction to Programming Language

Chapter 1 covers the history of programming language. Included in this chapter are types of programming languages and it generations. In addition, this chapter also covers various language translators.

Chapter 2 : Problem Solving Methods

Chapter 2 explain about phases revolve in Programming Development Life Cycle (PDLC). Additionally it also covers about concept of problem solving and different types and pattern in algorithm to solve problem.

Chapter 3 : Fundamentals of Programming Language

Chapter 3 covers about data, identifier, variable and constant. It also discuss about various operators like assignment, logical, increment, decrement, arithmetic and relational. Furthermore, this chapter also explains about control structures for sequence, selection and looping by utilizing flowchart and pseudocode.

Chapter 4 : Basic Programming Codes

Chapter 4 covers the element of the programming language, its standards, best practices in programming and steps in creating a C++ program. In addition, it also covers the basic programming concepts. Finally, this chapter also includes a topic on constructing a pseudocode and flow chart based on a given program code.

INTRODUCTION

This book is written with the intention of providing a quick reference for educators and students in problem solving and program design. Infographic concepts are applied to simplify and enhance readers understanding of the subject.

This book provides quick reference and information for problem solving concepts with various activities and exercises. Upon completion of this book, students will be able to solve a programming problem for a different scenarios.

Disclaimer:

Images used is modified from the Internet and free PowerPoint template. The originals are solely held by their owner.

Table of Contents

CHAPTER 1

Introduction to Programming Language

Photo by freddie marriage on Unsplash

Contents :

- History of programming language and approaches.
- Types of programming languages : Machine, Assembly, High Level
- Generations of programming language.
- Definitions of programmer, program and programming.
- Explain the language translators : Assembler, Compiler, Interpreter

Computer Fundamentals

- Electronic device.
- Machine Capable.
- Device.

Programming Language

- Set of rules
- Artificial language
- Example : C++, PHP, Java

QUICK FACTS

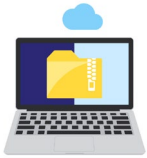
Generations

- Machine
- Assembly
- High Level

Translator

- Assembler
- Compiler
- Interpreter

1.1 Computer Fundamentals



Electronic Device

- Used to process data
- Simply a calculator, which works automatic and quite fast.

Machine Capable

- Machine capable of solving problems and manipulating data.

Device

- May define computer as a device that transform data.

How it Works?

There are variety electronic device now that act as computer such as smartphone, tablet, laptop that comes in variety of brand.

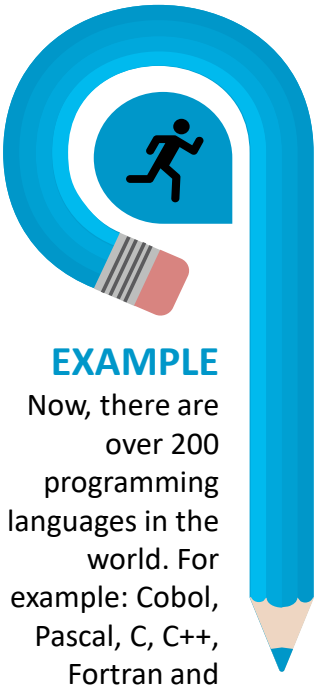
1.2 Programming Language

SET OF RULES

A set of rules and reserved words (keywords) that can be used to tell a computer what are the operations to be done.

ARTIFICIAL LANGUAGE

Artificial language composed by a fixed vocabulary and set of rules is used to create instructions for the computer to follow.



EXAMPLE

Now, there are over 200 programming languages in the world. For example: Cobol, Pascal, C, C++, Fortran and Java.

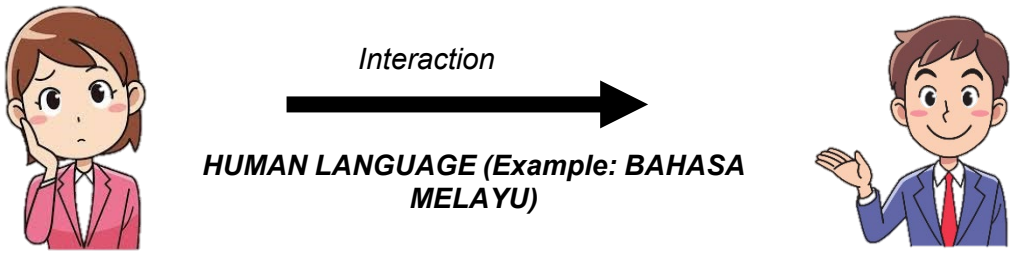
How it Works?

Current programming language around the world are Python, Javascript, Java, PHP, C#, C++, Ruby, Go, Dart Language, Visual Basic, Haskell, SQL and so on.

1.2 Programming Language

How Human Interact

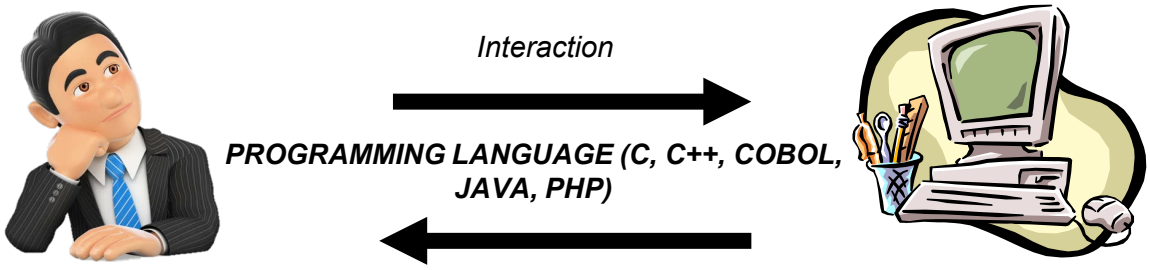
Example: People communicate using human languages, such as Bahasa Melayu, English and Japanese.



Human, Computer Interaction

How Human and Computer Interact

Example : People communicate with computers using programming languages such as C, C++, Cobol, Java and PHP.



1.3 Generations of Programming Language

1st Gen. Machine Language (1950-1960)

- Lowest level of language.
- Represented data and program instructions as 1s and 0s – binary digits.
- The ONLY language the computer could understand.
- Machine-dependent.

2nd Gen. Assembly Language (1950s)

- Known as symbolic language.
- Use mnemonic codes.
- Use assembler to translate the program to machine language.
- Machine-dependent.

3rd Gen. High Level Language (start 1960s)

- Called a procedural language.
- Written as a series of English-like words.
- Use translator (compiler or interpreter) to translate the program to machine language.
- Not machine-dependent.

4th Gen. High Level Language (start 1960s)

- Known as query language.
- Written as a series of English-like words.
- Use to retrieve data from database.
- Use translator (compiler or interpreter) to translate the program to machine language.

5th Gen. High Level Language (start 1960s)

- Known as knowledge-based language.
- Written as a series of English-like words.
- Programming language that resembles human language.

1.4 Generations of Programming Language

Advantage & Disadvantage

1st Gen. Machine Language (1950-1960)

- **Advantage :**
  - most efficient in terms of storage area and speed.
- **Disadvantage :-**
  - instruction and syntax in a program is not standardized.
- Machine-dependent. Example: a program written in Apple PC cannot be run in IBM PC.

2nd Gen. Assembly Language (1950s)

- **Advantages:-**
  - Program can run faster.
  - An efficient use of the memory.
- **Disadvantages:-**
  - Errors can be made easily.
  - Difficult to understand.
  - It takes a long time to write the program.

3rd Gen. High Level Language (start 1960s)

- **Advantage:-**
  - Machine-Independent. The program can be written and executed on any computer.
  - Easy to be developed and maintained.

4th Gen. High Level Language (start 1960s)

- **Advantage:-**
  - Machine-Independent. The program can be written and executed on any computer.
  - Easy to be developed and maintained.
  - It is simpler and easier to code.

5th Gen. High Level Language (start 1960s)

- **Advantage:-**
  - Machine-Independent. The program can be written and executed on any computer.
  - Easy to be developed and maintained.

1.5 Examples of Programming Language

Example : Machine language

```
0100
110010    100000    1111000    100000    110100
111101    100000    100000
11110
10111000
1010000
11101000
11111100
10111000
10001110  11011000
10111000
10001110  11000000
```

Instruction to compute 2 * 4

Example : Mnemonic code

```
Assembly Language (Mneumonic Code)

sseg                                  segment   stack
                                      db 256    dup  (?)

sseg                                  ends
dseg                                  segment
data                                  db "2 x 4 =  "
dseg        ends
cseg                                  segment
assume      cs:cseg,ds:sdeg,ss:sseg,es:nothing
start       proc far
                                      push ds
                                      mov ax,0
```

Instruction to compute 2 * 4

Example : BASIC program

```
'BASIC PROGRAM
'AVERAGING THREE INTEGERS
PRINT "Mengira Purata"
PRINT "--------------------"
PRINT "Nilai A = "; 20
PRINT "Nilai B = "; 30
PRINT "Nilai C = "; 40
PRINT "Purata tiga nombor
ialah "; (20+30+40)/3
END
```

Instruction to compute average of 3 numbers

Example : SQL Language

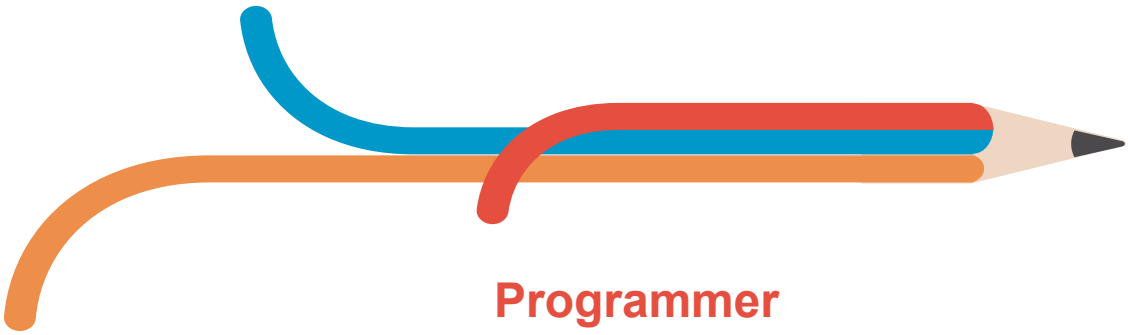
```
Create table order

(OWNERID INT(5) NOT NULL
UNIQUE,

  ITEM CHAR (40) NOT NULL);
```

Instruction to compute average of 3 numbers

1.6 Definition of Program, Programming, Programmer

Program

- A set of step-by-step instructions that directs a computer to perform a specific task and to produce the required results.
- It's written in a programming language and is converted into the computer's machine language.



Programmer

- Person who writes the program.

Programming

- Programming is a process of designing / creating a program.

1.7 Language Translator

- It is a program that will checks **syntax** of a program to ensure the programming language is used correctly, by giving you the syntax-error messages, which is known as **diagnostics**.

- Translates a program into language that the computer can understand.
- Produces descriptive error messages known as **syntax-error** to programmer if there are errors in the programs.

- **Syntax** – set of rules to be followed when writing a program, these rules are similar to the grammatical rules in English.
- **Example:**
  Message displayed on the screen is “please enter a number”.

  C++ Language : cout<<“please enter a number”;

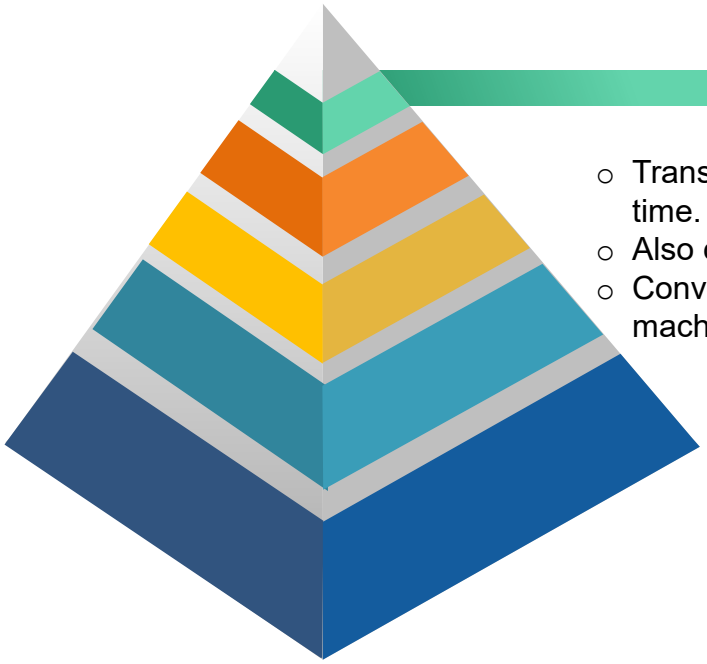
- **Syntax Error**– error in the use of a programming language.
- **Example in C++:**
  Statement : system("pause")

  If semicolon (;) is not placed after statement, a message will display:

  “**error C2143: syntax error : missing ';' before '}‘** ”

  The correct syntax is : system("pause");

1.8 Types of Translator



ASSEMBLER

- Translates the whole program at one time.
- Also called assembler program.
- Convert assembly language to machine language.

```
Sseg        segment
stack
db 256      dup (?)

Sseg       ends

dseg       segment
data       db "2 x 4 = "
dseg ends
cseg       segment
Assume

cs:cseg:ds:sdeg:ss:sseg
es:nothing
Start      proc far
```

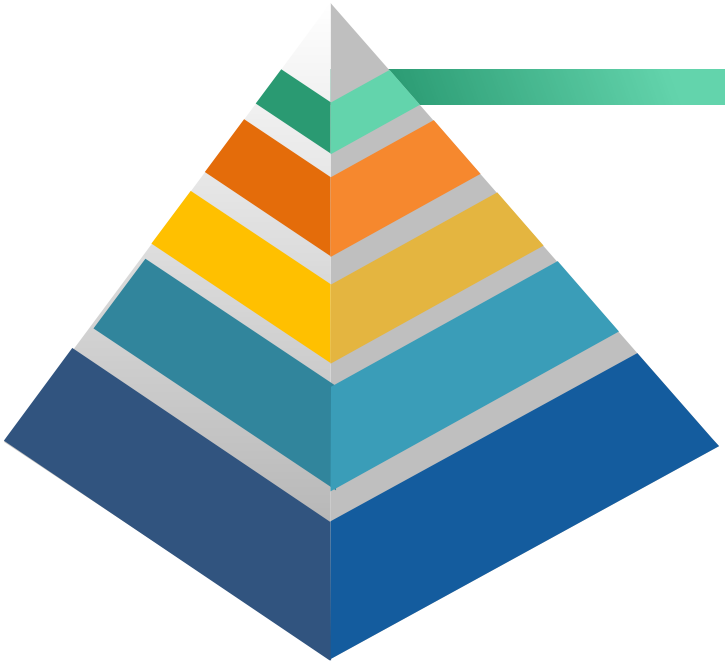
Assembly Language

→ ***Assembler*** →

```
11101      100000
           100000
11110
10111000
11101000
11111100
10111000
10001110   11011000
10111000
10001110   11000000
```

Machine Language

1.8 Types of Translator



COMPILER

- Translates the whole program at one time.
- Translates the source code, for COBOL language to machine example language.
- Used to translate C, COBOL and other language.
- A program that translates from a low level language to a higher level one is a *decompiler.*

```
# include <stdio.h>

int main( )
{
    int x, y, z, sum;
    double avg;
    printf(" Enter 3 numbers:");
    scanf("%d %d %d",&x, &y, &z);
    sum = x + y + z;
    avg = sum / 3;
    printf("x = %d y = %d z = %d",
    x, y, z);
    printf("the average is = %f",
    avg);
    return 0;
 }
```

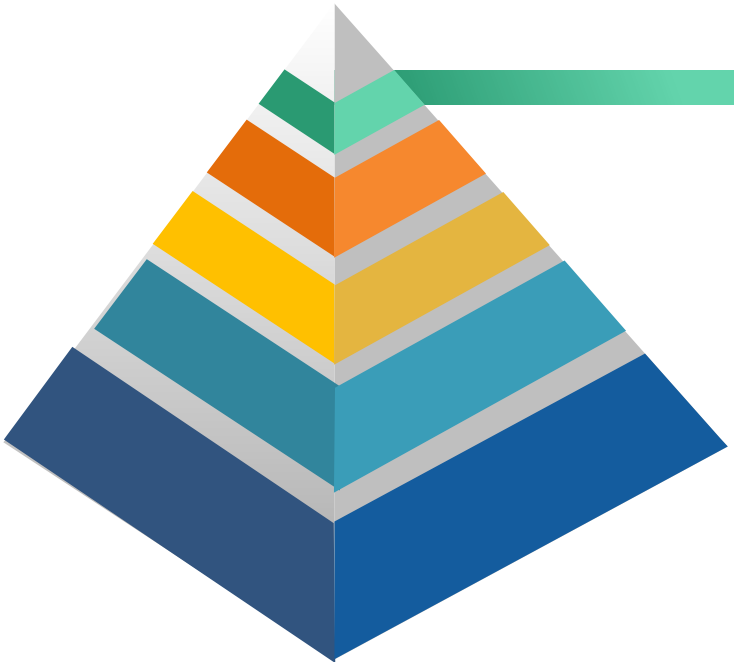
C Language

→ ***Compiler*** →

```
11101     100000    100000
11110
10111000
11101000
11111100
10111000
10001110  11011000
10111000
10001110  11000000
```

Machine Language

1.8 Types of Translator



INTERPRETER

- Program that translates a high level language to machine language.
- It translates one line at a time and executes each line after it is translated.
- Displays feedback as the error occurred.
- Used to translate BASIC and SQL language.

```
'BASIC PROGRAM
'AVERAGING THREE
INTEGERS
PRINT "Mengira Purata"
PRINT "---------------------"
PRINT "Nilai A = "; 20
PRINT "Nilai B = "; 30
PRINT "Nilai C = "; 40
PRINT "Purata tiga nombor
ialah ";      (20+30+40)/3
END
```

Basic Language

→ ***Interpreter*** →

```
10001110  11000000
10111010
10111011
10001101  110110
10111111
10110000  000000010
10110011  000001000
11100011  111111000
00001110  110000
10100000
10111001
10100100
```

Machine Language

Activity 1.1

1. Write the definitions of :
   a) Programmer
   b) Programming
   c) Program
   d) Language translator

Answer

Words	Definitions
a) Programmer	Person who writes the program.
b) Programming	Process of designing/ creating a program.
c) Program	step-by-step instructions that directs a computer to perform a specific task and to produce the required results.
d) Language translator	program that will checks syntax of a program to ensure the programming language is used correctly, by giving you the syntax-error messages, which is known as diagnostics.

Exercise 1.1

1. List and explain generations of programming language.
2. State translator that translates high level language to machine language.
3. Differentiate between compiler and interpreter.

Answer

1.

2.

3.

Compiler	Interpreter

CHAPTER 2

Problem Solving Methods

Contents :

- Phase in Programming Life Cycle (PLC)
- Problem solving concept.
- Input, process and output.
- Identify the input, process and output based on problem statement.
- Different types and pattern in algorithm to solve problem.

QUICK FACTS

Programming Life Cycle

- Specify the problem
- Analyze the problem
- Design algorithm
- Implement algorithm
- Test and verify
- Maintain and update
- Documentation

Problem Analysis

- Input
- Process
- Output

Patterns in algorithm

- Sequential
- Conditional
- Iterational

Flowchart symbols

- Terminator
- Connector Lines
- Process
- Input / Output
- Decision
- On and Off Page Reference

2.1 Phases in Programming Life Cycle (PLC)

1

Specify the problem

To specify the inputs and outputs, processing requirements, system constraints, and error handling methods

2

Analyze the problem

To review program specifications, meeting with the analyst and users and identifying program components. Precisely define the problem to be solved and write program specifications.

3

Design algorithm to solve problem

Develop a detailed logic plan using a tool such as pseudocode, flowcharts, object structure diagrams, or event diagrams to group the program's activities into modules. Plan a method of solution or algorithm for each module; and test the solution algorithms.

4

Implement the algorithm

Translate the design into an application using a programming language or application development tool by creating the user interface and writing code.

5

Test and verify the completed program

Test the program, finding and correcting errors (debugging) until it is error free.

There are three (3) types of error occur during this phase which are syntax error, logic error and runtime error.

6

Maintain and update the program

Provide education and support to end users. Correct any unanticipated errors that emerge and identify user-requested modifications (enhancements).

7

Documentation

Review and if necessary, revise internal documentation; formalize and complete end-user (external) documentation

Activity 2.1

Apply the first 2 phase of PLC for the case study below:

Problem:

Annie wishes to bake a cake. But all the ingredients are weight is ounces. Please help Annie to convert the measurement in grams which is 1 ounces is equal to 28.4395g .

Answer

1st phase - Specify the problem.

In this phase, the development team visits the customer and studies their system. The essential purpose of this phase is to find the need and to define the problem that needs to be solved.

2nd phase - The development team will identify the types of cake to bake. Then check the ingredients need to bake a cake. They also studies about measurement and equipment's need to use to bake a cake.

Exercise 2.1

Identify the phase involve for the activities below:-

1. Develop a detailed logic plan.
2. This step is also known as programming phase.
3. Syntax error, logic error and runtime error always occur in this phase.

Answer	
1.	
2.	
3.	

2.2 Problem Solving

Problem Solving

- Thinking process and is part of the larger problem process that includes problem finding and problem shaping.
- Occurs when a system needs to move from a given state to a desired goal state.
- A computer cannot solve a problem on its own. Programmer has to provide step by step solutions which the computer can understand and execute.
- Different people use different techniques to solve problem.
- To be a good problem solver and good programmer, you must follow good problem-solving techniques.

Notes :

Problem analysis is to describe in detail a solution to a problem and information needed in solving the problem.

Problem Solving Step

- Identify the inputs and outputs
- Identify other data and constants required to solve the problem.
- Identify what needs to be computed.
- Write an algorithm.

2.3 Method for defining problem using IPO

Input

Process of entering data and programs in to the computer system

Process

Task of performing operations like arithmetic and logical operations

Output

Process of producing results from the data for getting useful information.

Reminder :

Keyword : entering data, input unit

Reminder :

Keyword : arithmetic and logical operation, CPU, Storage Unit

Reminder :

Keyword : producing result, useful information

Input

Process

Output

Activity 2.2

- Assume that the transaction is money withdrawal. The instructions are below. Identify the IPO for each statements.
  - » Get the card number from the user
  - » Get pin number from the user
  - » Process the input data
  - » Get the transaction chosen by the user
  - » Get the account type from the user
  - » Process the transaction as wanted by the user
  - » Withdraw amount of money required by the user
  - » Print receipt for the user

Answer

Input :

- » Get the card number from the user
- » Get pin number from the user
- » Get the transaction chosen by the user
- » Get the account type from the user

Process :

- » Process the input data
- » Process the transaction as wanted by the user

Output :

- » Withdraw amount of money required by the user
- » Print receipt for the user

Exercise 2.2

Discuss on steps needed to deposit cash on Cash Deposit Machine (CDM). Identify input, process and output.

Answer

Input :

Process :

Output :

Activity 2.3

Write a program that will get 3 numbers as input from the users. Find the average and display the three numbers and its average.

Answer

Problem Analysis:

Input :

number1, number2, number3.

Process:

1. Add the three numbers using formula:

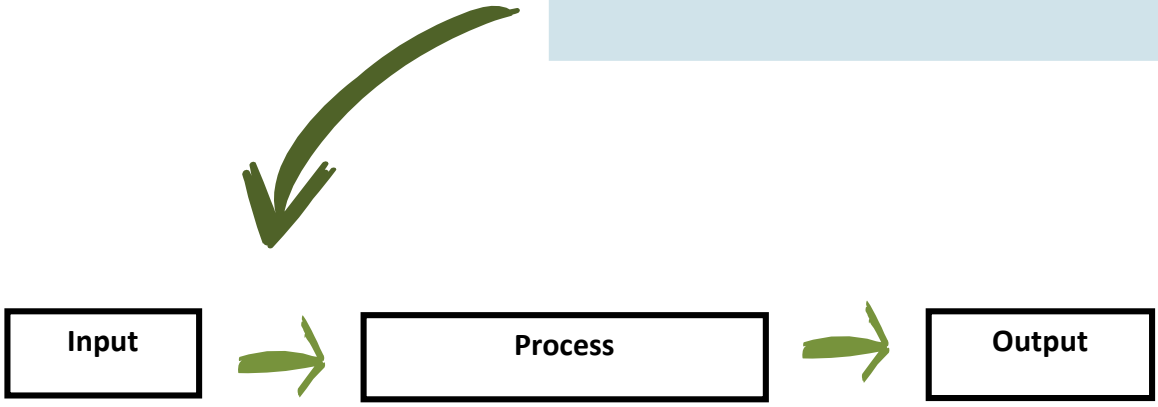
   sum = number1 + number2 + number3

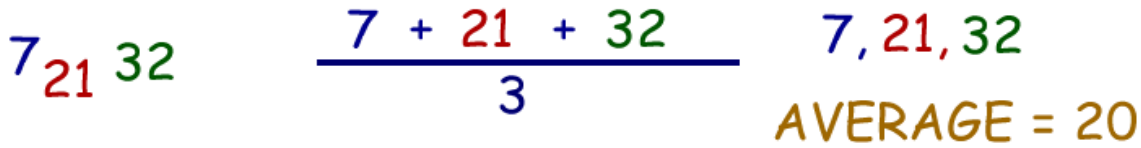
2. Calculate average using formula:

   average = sum / 3

Output:

number1, number2, number3 and average





Activity 2.4

How fast is a car traveling if it goes 50 miles in 2 hours?

Answer

IPO Chart

Input	Process	Output
distance, timeHour	speed = distance/time Hour	speed

Answer

Problem Analysis:

Input :

distance and timeHour

Process:

Calculate speed using formula :

speed= distance / timeHour

Output:

speed

Activity 2.5

Calculate the total fine charged by library for late return books. The charge is RM0.20 for 1 day

Answer

IPO Chart

Input	Process	Output
day	total_fine= day x fine	total_fine

Answer

Problem Analysis:

Input :

day

Process:

1. Set fine = 0.20
2. Calculate total fine charge using formula :

   total_fine= day x fine

Output:

total_fine

Exercise 2.3

Uncle Ahmad wants to buy 5 cans of paint to paint his house. The price for a can of paint is RM 25.50. Calculate the price to be paid for 5 cans of paint that bought.

Answer

IPO Chart

Input	Process	Output

Answer

Problem Analysis:

Input :

Process:

Output:

Exercise 2.4

Calculate and display the total mark of three quizzes, where the users have to enter the mark of each quiz.

Answer

IPO Chart

Input	Process	Output

Answer

Problem Analysis:

Input :

Process:

Output:

01

LOGIC

- ❑ Reasoning conducted or assessed according to strict principles of validity.
- ❑ A system in a computer or electronic device so as to perform a specified task.

02

LOGIC and ALGORITHM

- ❑ Systematic logical approach which is a well-defined, step-by-step procedure that allows a computer to solve a problem.

03

PROGRAM DESIGN

- ❑ It is a framework or flow that shows the steps in problem solving.
- ❑ some organizational tools to help us solve problems on the computer such as the following.
  - Flowchart
  - Pseudocode

2.4 Algorithm

01

A sequence of instructions to solve a problem, written in human language.

02

A step-by-step procedure to solve a given problem.

04

Programmer writes the problem solving in the form of an algorithm before coding it into computer language.

03

An algorithm should always have a clear stopping point.

05

Algorithms for making things will divided into sections;

- The parts/components/ Ingredients (**inputs)** required to accomplish the task
- actions/steps/methods (**processing)** to produce the required outcome(output).

06

Algorithm can be developed using:

- Pseudo code (Similar to programming language)
- Flow chart

2.5 Qualities of good Algorithm

01 Inputs and outputs should be defined precisely.

02 Each steps in algorithm should be clear.

03 Algorithm should be most effective among many different ways to solve a problem.

04 An algorithm shouldn't have computer code.

05 Algorithm should be written in such a way that, it can be used in similar programming languages.

Activity 2.6

An example of algorithm and life for people use to make a recipe for a cake.

Answer

1. Heat the oven at 180 Celsius.
2. Prepared the baking tin (mould).
3. Beat the butter and sugar.
4. Add eggs and vanilla essence.
5. Stir in the flour into the mixture.
6. Pour into a mould.
7. Put in the oven.

Activity 2.7

An example of algorithm and life for student to attend a lecture in faculty.

Answer

1. wake up
2. preparation to attend class
3. heading to the faculty.

Identified the main steps, do the refinement algorithm for each step. Step 1,2 and 3 may be refined or detail.

Answer

Refinement step 1: wake up

1. rise out of bed at 6 am
2. brush the tooth and take bath
3. dawn prayer
4. make up bed

Refinement step 2: preparation to attend class

1. Breakfast
2. Dressing
3. wear shoes

Refinement step 3: heading to the faculty.

1. walk to the nearest bus stop
2. waiting for bus
3. boarding a bus
4. go to the lecture hall

Activity 2.8

An example of algorithm and life for preparing a cup of coffee.

Answer

1. boil water
2. add a cup of coffee
3. if necessary sugar
   3.1. add sugar
4. if necessary cream
   4.1 add cream
5. put hot water into the cup

Exercise 2.5

Write a program that will get 3 numbers as input from the users. Find the average and display the three numbers and its average.

Answer : Problem Analysis

Answer : Algorithm

Exercise 2.6

Write a program to calculate the area of a circle.

Answer : Problem Analysis

Answer : Algorithm

2.6 Patterns in Algorithm

A

Sequential

(Sequence structure):

1. step_1
2. step_2
3. step_3

:

n. step_n

B

Conditional

(The structure of the selection):

if condition then

code block true

else

code block is not true

End if

C

Iterational

(Repetition/Looping):

repeat

part to repeat

until condition

Example Sequential

1. Set total = 0
2. Input quiz1, quiz2
3. Calculate total quiz :

   total = quiz1 + quiz2
4. Display total

Example Conditional

1. Enter testMark
2. Check testMark

   2.1 If testMark >= 50

   Display "PASS"

   2.2 If not

   Display "Try Again"
3. Display statement

Example Iterational (do..while)

1. Set no=0; Average = 0; Total = 0
2. Input number
3. Add Total using formula:
   - Total = Total + number
4. Add Counter using formula:
   - no = no + 1
5. Compare whether condition is less than 5
   - If yes , go to step 2
   - If not, go to step 6
6. Calculate Average of numbers using formula;
   - Average = Total/5
7. Display Average

2.7 Flowchart

Definition

Flowchart is a **graphical representation** of data, information and workflow using certain symbols that are connected to flow lines to describe the instructions done in problem solving.

Using flowchart, programmer can **test the solution** for bugs and go on to code the solution to the problem into a computer language for entry into the computer.

Flowchart is a **tool developed** in the computer industry, for **showing the steps involved in a process.**

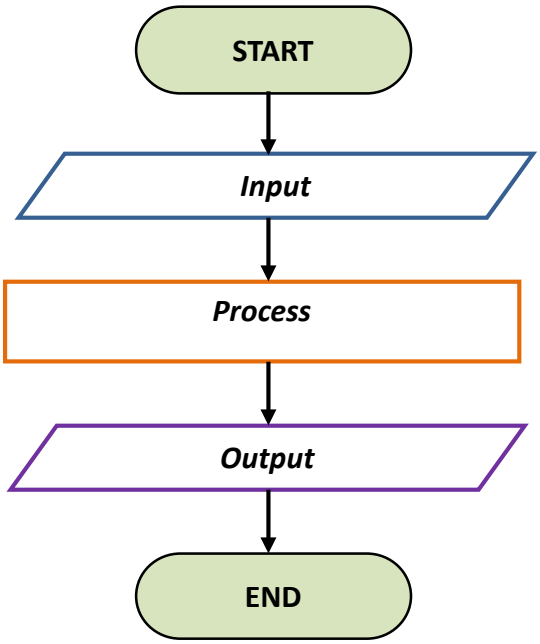
2.8 Flowchart symbol

Symbol	Name	Function
Start End	Terminator	Indicates the starting or ending the program, process or interrupt program.
	Connector lines	Shows direction of flow.
	Process	Indicates any type of internal operation inside the processor or memory.
	Subroutine process/functions	Used to invoke a subroutine or an interrupt program.
	Input / Output	Used for any input / output operation. Indicates that the computer is to obtain data or output result.
	Decision	Used to ask question that can be answered in binary format (Yes/No, True/False)
A A	Connector	At same page
2 3	Connector	Off-page connector

2.9 Flow of process using flowchart symbol

Notes :

It shows the flow of the process from the start to the end of the problem solving.



Entering data from keyboard / input device

Task performing operation / formula

Display data

2.10 General rules for flowcharting

1. All boxes of the flowchart are connected with Arrows. (Not lines)

2. Flowchart symbols have an entry point on the top of the symbol with no other entry points. The exit point for all flowchart symbols is on the bottom except for the Decision symbol.

3. The Decision symbol has two exit points; these can be on the sides or the bottom and one side.

4. Generally a flowchart will flow from top to bottom.

5. Connectors are used to connect breaks in the flowchart. Examples are:
   - From one page to another page.
   - From the bottom of the page to the top of the same page.
   - An upward flow of more then 3 symbols

6. Subroutines and Interrupt programs have their own and independent flowcharts.

7. All flow charts start with a Terminal or Predefined Process (for interrupt programs or subroutines) symbol.

8. All flowcharts end with a terminal or a contentious loop.

Activity 2.9

Convert algorithm into flowchart.

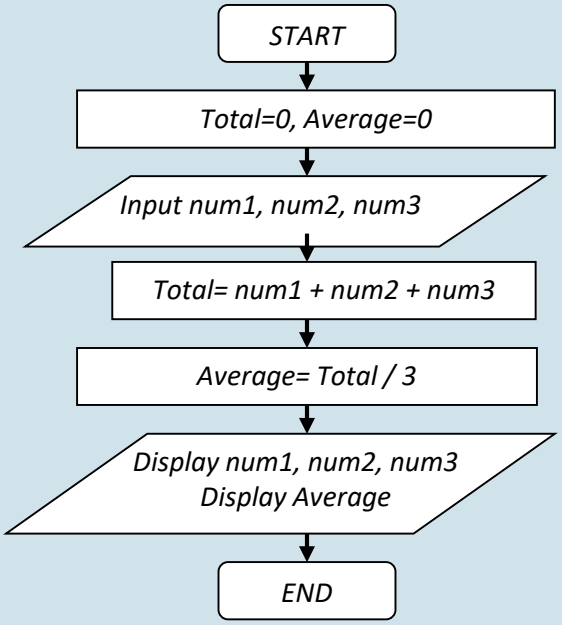
Algorithm

1. Set Total=0, Average=0;
2. Input num1, num2, num3
3. Total up the 3 numbers

   Total= num1+num2+num3
4. Calculate average

   Average=Total/3
5. Display num1, num2, num3 and the average

Answer : Flowchart



Exercise 2.7

Convert algorithm given into flowchart.

Algorithm

1. Set Pi=3.142
2. Input a radius value.
3. Calculate area using formula:

   area = Pi * radius * radius
4. Print the area of the circle.

Answer : Flowchart

2.11 Pseudocode

Definition

Steps in problem solving, written half in programming code and half in human language.

Semi-formal, English-like description of logic of the algorithm.
For example, some parts uses C language code and some parts use Malay or English language.

Helps programmer to "think out" the program.
Advantage: Simple and easy to understand

Criteria of good pseudocode

- Eventually ends.
- Gives the correct solution in all cases.
- Easy to understand, precise and clear.

Format pseudocode

```
START
    Input
    Statement
    Output
END
```

2.12 How to write Pseudocode statement

SIX(6) basic computer operations.

1

Computer can receive information

- Read (information from a file)
- Get (information from the keyboard)

2

Computer can put out information

- Write (information to a file)
- Display (information to the screen)

3

Computer can perform arithmetic

- Use actual mathematical symbols or the words for the symbols
- Example:
- Add number to total
- Total = total + number
- +, -, *, /
- Calculate, Compute also used

4

Computer can assign a value to a piece of data

Three (3) cases :

i. To give data an initial value (using Initialize, Set)

ii. to assign a value as a result of some processing
- "="
- *x=5+y

iii. to keep a piece of information for later use
- Save, Store

5

Computer can compare two piece of information and select one of two alternative actions.

```
IF condition THEN
   some action
ELSE
   alternative action
ENDIF
```

6

Computer can repeat a group of actions

```
WHILE condition (is true)
            some action
ENDWHILE

FOR a number of times
            some action
ENDFOR
```

2.13 Rules to write pseudocode

1. Each step must be clear and easy to understand.
2. Each step not more than 2 actions.
3. The steps are implemented sequentially.
4. Each step consists of keywords that describe the operation / action.
5. Statements in sequence can be grouped into blocks consisting of start and end keywords.
6. Each step is written in a separate row, if necessary, the next line must be indent.

Activity 2.10

Convert algorithm into pseudo code to calculate area of circle.

Algorithm

1. Set Pi = 3.142
2. Input radius value.
3. Calculate the area using the given formula:

   area = Pi * radius * radius
4. Print the area of the circle.

Answer : Pseudocode

```
START
    Pi = 3.142
    Input radius
    area = Pi * radius *radius
    Output area
END
```

Exercise 2.8

Convert algorithm into pseudo code to calculate the average of 3 numbers.

Algorithm

1. Set Total=0, Average=0
2. Input a, b, c
3. Total up the 3 numbers

   Total= a+b+c
4. Calculate average

   Average=Total/3
4. Display the average

Answer : Pseudocode

Exercise 2.9

Write problem analysis, algorithm, pseudocode and flowchart for scenario below :
You had bought a nice shirt which cost RM29.90 with 15% discount. Count the net price for the shirt.

Answer : Problem Analysis

Answer : Flowchart

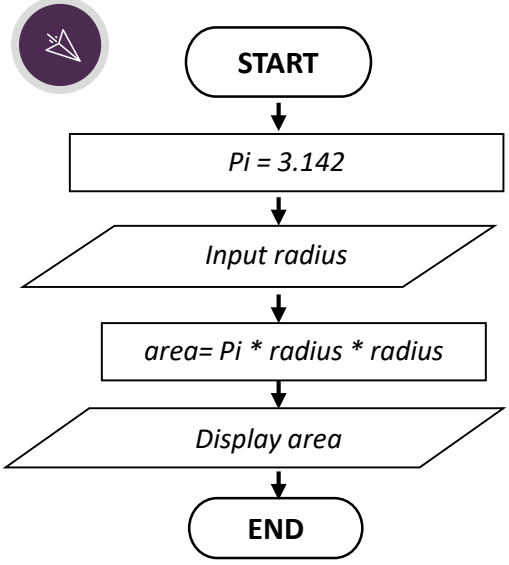
Answer : Algorithm

Answer : Pseudocode

2.15 Pseudocode vs Flowchart

Pseudocode	Flowchart
• Steps in problem solving that is written half in programming code and half in human language	• A graphical representation of instructions done in problem using certain symbols that are connected to flow lines
• Work well for small problems	• Used for larger problems
• Which a mixture of English statement , some mathematical notation, selected keywords from programming language	• Diagrammatic method which consists of terminator boxes, process boxes and decision boxes, with flow of logic indicated by arrows

```
START

    Pi = 3.142
    Input radius

    area = Pi * radius *radius

    Output area

END
```



CHAPTER 3

Fundamentals of Programming Language

Photo by Aleks Dorohovich on Unsplash

Contents :

Data

- Data is an array of facts that can be modified by computer into useful form for human.
- Users deal with data in ways that let them know what kind of information they are managed. Data is managed by the instructions in computer language.
- Examples of data are number, text, currency and others.

Data Types

- A data type is a classification of the type of data that a variable or object can hold in computer programming.
- A data storage format that can contain a specific type or range of values.
- When computer programs store data in variables, each variable must be assigned a specific data type.
- Examples of data types include integers, floating point numbers, characters, strings, and arrays.

Identifier

- Identifiers are used to name constants, variables, function names and labels.
- Identifiers are formed by combining letters (both upper and lower case letters) digits and the underscore (_).
- Identifiers in C++ are case sensitive.

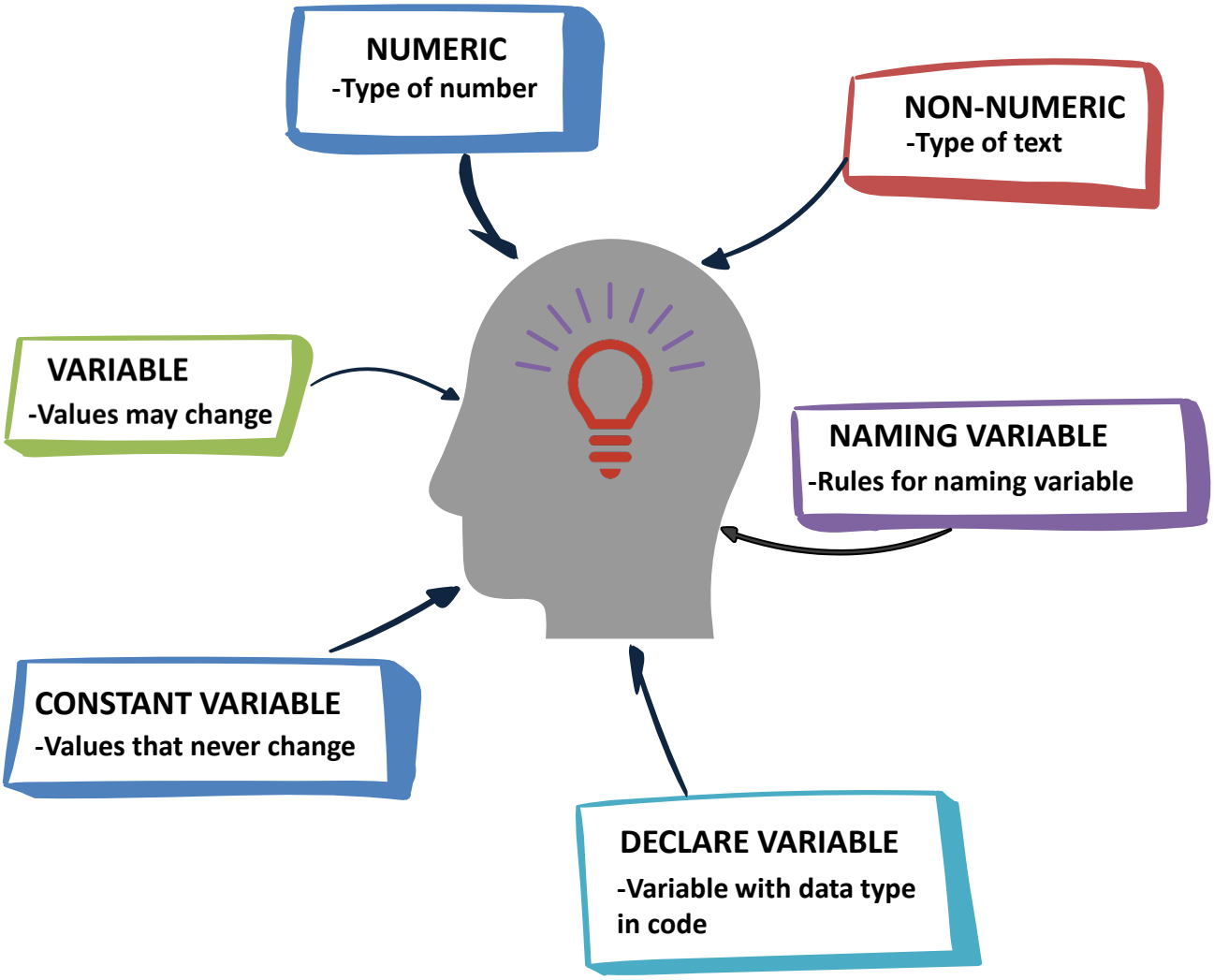
Variable

- A variable is any measured characteristic or attribute that differs for different subjects.
- Variables also can be defined as identifiers whose value may change during the course of execution of a program. Each data used for computer program is stored in a variable.

Constant

- Constants are used to store values that never change during the program execution.
- Using constants makes programs more readable and maintainable. *Constants* are expressions with a fixed value.

3.1 Data & Identifier a big picture view



3.2 Major Of Data

There are two major types of data which are:

NUMERIC

01 - INTEGER

- Positive and negative numbers including zero and no decimal place.
- Keyword : **int**
- Example: **0, +1, -10.**

02 – REAL NUMBER

- The number will be stored in floating point.
- Keyword : **float**
- Example: **36.7C, 234.55kg, 3.142** Keyword : **double**
- Example: **RM12.0019, RM1020.0009**

NON-NUMERIC

03 - CHARACTER

- Consists of all letters, numbers and special symbols.
- Characters are surrounded by single quotation mark (‘ ‘).
- Example: ‘A’, ‘m’,’=’, ‘#’, ‘1’ or ‘ ‘

04 - BOOLEAN

- Used in making yes-or-no decisions (TRUE-or-FALSE).
- Assume a = 2 and b = 5

```
If (a < b)
    Decision is TRUE
EndIf
```

05 - STRING

- Combination of more than one character.
- Keyword : **string**
- Example: **“Politeknik” , “IPTA”**

3.3 Naming Variable

RULES
Naming Variable

1. Must begin with a letter or underscore (_) symbol.

2. Spaces between two terms are not allowed.

3. Identifier cannot more than 31 characters.

4. Identifier names cannot be the same as the **RESERVED WORD**

5. Case sensitive

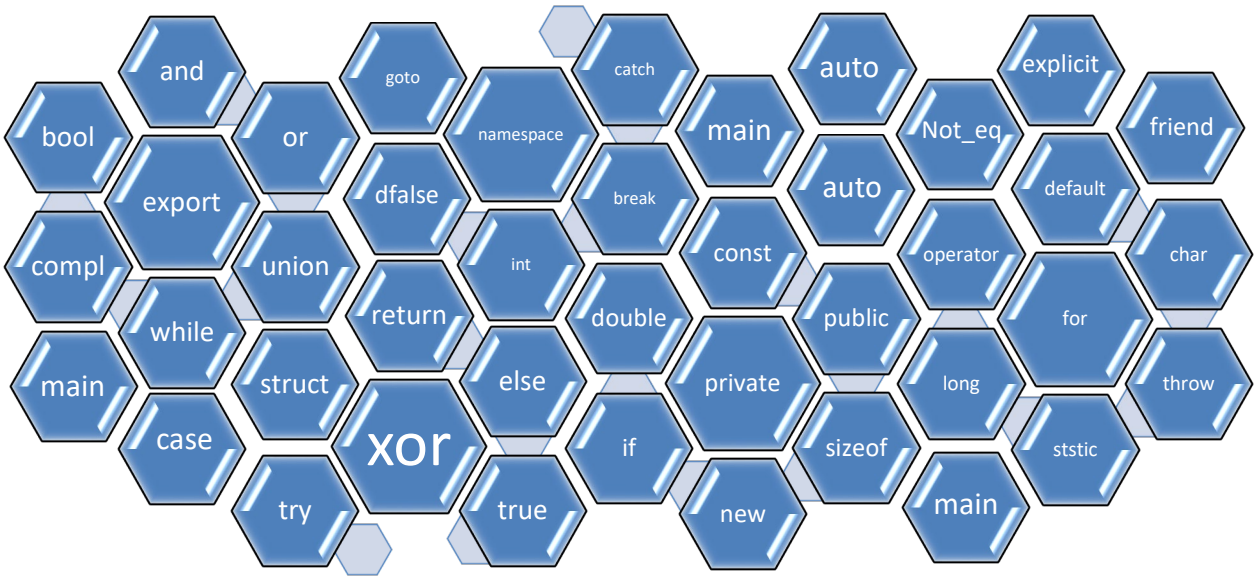
   Example:
   Variable name "MyNumber" Is different with "MYNUMBER"

6. Special character such as : *,@,#,!,$,%,^,&,(,) are not allowed.

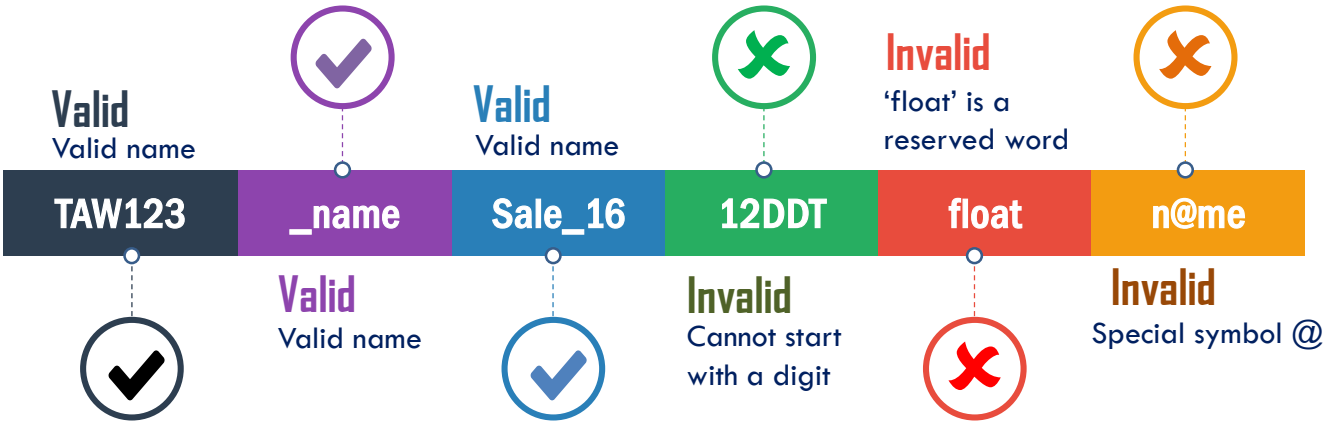
3.4 Reserve Word

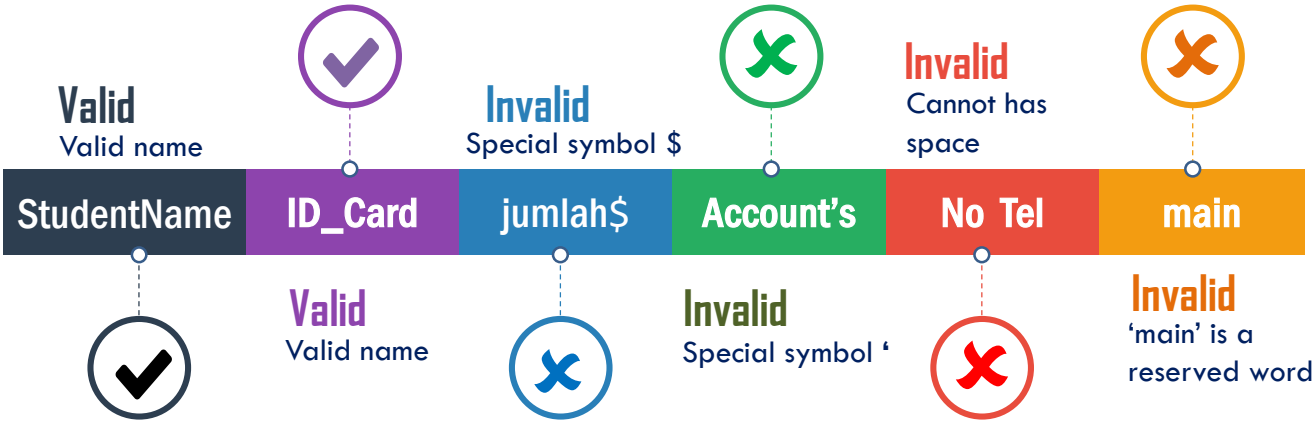
Reserve Word

- a word in a programming language that has a fixed meaning and cannot be redefined by the programmer.
- a word that cannot be used as an identifier, such as the name of a variable, function, or label, it is "reserved from use“
- A special word reserved by a programming language or by a program.



3.5 Valid or Invalid Identifier





3.6 Declaration Variable and Constant

Syntax to Declare a Variable and Constant

Syntax: **data type variable ;**
float weight ;
float weight =55.5;

Syntax: **const data type variable_name**= value;
const int boiling = 100;
#define boiling 100;

Declare a Variable

int

int age;
int number;

char

char Grade;
char name[10];

float

float height;
float salary;

Constant double

const double PI = 3.142;
#define PI 3.142;

Activity 3.1A

1. Identify and write either valid or invalid for the variables below.
   a) float hasilBahagi;
   b) int 2jumlah;
   c) double Jalan#2;
   d) int pertama;
   e) float hasil_tambah;

Answer	Valid (/) / Invalid (x)	Reason
a) float hasilBahagi;	Valid	-
b) int 2jumlah;	Invalid	Cannot with digit
c) double Jalan#2;	Invalid	Cannot use special symbol '#'
d) int pertama;	Valid	-
e) float hasil_tambah	Valid	-

Activity 3.1B

1. Declare variables for the statements below.
   a) Keep the average mark of students in a class.
   b) Keep the grade of students occupying for a computer course.
   c) Keep the CGPA results for the students.
   d) Store the value of PI to 3.142.
   e) Keep the age of worker in the system.

Answer	Declaration
a) Keep the average mark of students in a class.	float mark;
b) Keep the grade of students occupying for a computer course.	Char Grade;
c) Keep the CGPA results for the students.	float CGPA;
d) Store the value of PI to 3.142.	const double PI=3.142;
e) Keep the age of worker in the system.	int worker_age;

Exercise 3.1A

1. Identify and write either valid or invalid for the variables below.
   a) return
   b) _balance
   c) my_politeknik
   d) average,mark1
   e) Stud Name
   f) $Salary

Answer	Valid (/) or Invalid (x)	Reason
a) return		
b) _balance		
c) my_politeknik		
d) average,mark1		
e) Stud Name		
f) $Salary		

Exercise 3.1B

1. Declare variables for the statements below.
   a) Total salary per month.
   b) Keep the temperature reading.
   c) Interview result either pass or not.
   d) Permanent value for color is a red.
   e) Count the number of catch fish.

Answer	Declaration
a) Total salary per month.	
b) Keep the temperature reading.	
c) Interview result either pass or not.	
d) Permanent value for color is a red.	
e) Count the number of catch fish.	

Exercise 3.1C

Problem 1 :
A simple program is created to store personal data for employee. The information that needs to be store includes full name, phone number and employee number. State the suitable variable, data type and complete declaration for the program.

Variable	Data Type	Declaration

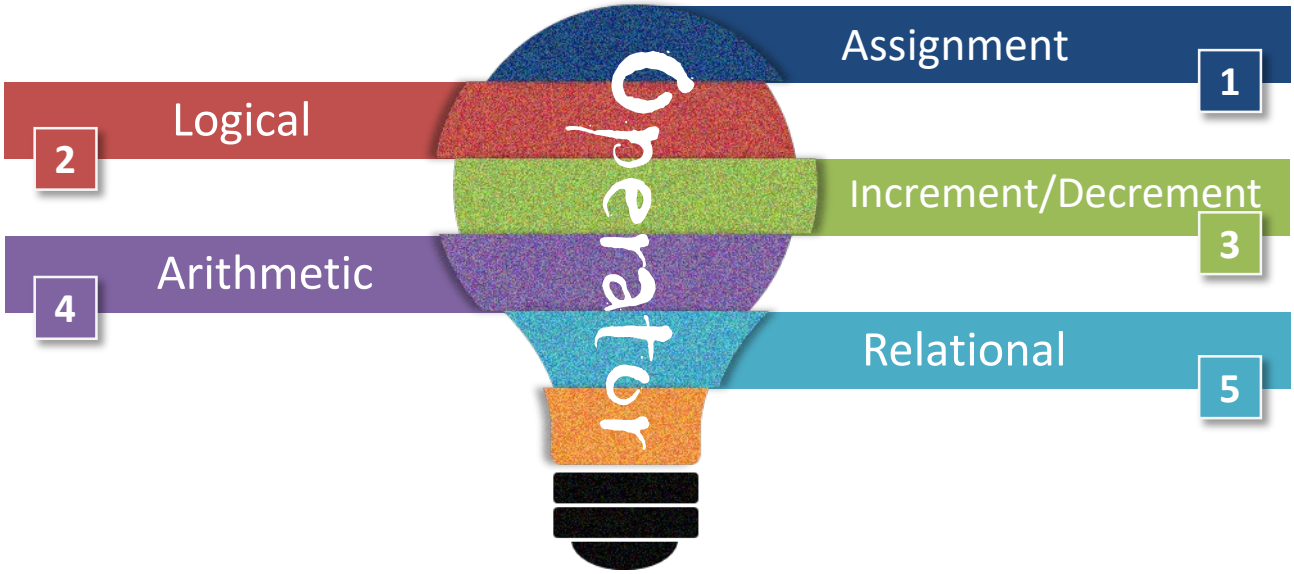
Problem 2 :
You are creating a simple program to identify if a student passed or fail for their marks in the mid semester test. State the suitable variable, data type and complete declaration for the program.

Variable	Data Type	Declaration

Problem 3 :
A simple program is created to calculate area of a circle using a given radius. State the suitable variable, data type and complete declaration for the program.

Variable	Data Type	Declaration

Contents :



Operator

- An operator is a symbol or character that causes the compiler to take an action.
- Example : '+' is an arithmetic operator that represent addition.
- Operators act on operands and all operands are expressions.

Operand

- Operand is the object that are manipulated;
- example:

  2 + y / x

  explain:

  2, *y* and *x* are operands.

3.7 Assignment, Logical, Increment& Decrement, Arithmetic, Relational

1

Assignment	Meaning	Expression
x = y	Direct assignment	x=y
x += y	Assignment by sum	x = x + y
x -= y	Assignment by difference	x= x - y
x *= y	Assignment by product	x= x * y
x /= y	Assignment by quotient	x= x / y
x %= y	Assignment by remainder	x=x % y
x &= y	Assignment by bitwise AND	x=x & y
x |= y	Assignment by bitwise OR	x=x | y
x ^= y	Assignment by bitwise XOR	x= x ^ y

2

Logical	Meaning	Expression x=1, y=0	
&&	AND	x && y	(False)
||	OR	x || y	(True)
!	NOT	! x	(False)

4

Arithmetic	Meaning	Expression x=10 ,y=3
++ , --	unary	-
*	Multiplication	Result =x * y; Result=30
/	Division	Result =x / y; Result=3.3
%	Modulus	Result =x % y; Result=1
+	Addition	Result =x + y; Result=13
-	Subtraction	Result =x – y; Result=7

highest

lowest

5

Relational	Meaning	Expression
==	equals	100 = =100
>	Greater than	100 > 50
<	Lower than	50 < 100
>=	Greater or equals	100 >=90
<=	Lower or equals	90 <=100
!=	Not equals	90 !=100

3

Increment /Decrement	Meaning	Expression x=15
++ x	Pre increment	y=++x ; y=16
x ++	Post increment	y=x++ ; y=15
- - x	Pre decrement	y=--x ; y=14
x - -	Post decrement	y=x-- ; y=15

Activity 3.2A

1. Based on the following expression problem, find the value and show the calculation steps.

   a) 4 + 2 * 3 – (5 / 2)
   b) 5 * 2 + 5 % 2
   c) 5 + 5 * (6 – 2)

	Answer
a) 4 + 2 * 3 – (5 / 2)	4 + 2 * 3 – (5 / 2) 4 + 2 * 3 – (2) 4 + 6 – 2 10 – 2 8
b) 5 * 2 + 5 % 2	5 * 2 + 5 % 2 10 + 5 % 2 10 + 1 11
c) 5 + 5 * (6 – 2)	5 + 5 * (6 – 2) 5+5*4 5+20 25

Activity 3.2B

Find the value for the following expression.
Given a = 4, b = 8;

a) !(a==b) && (a>b)
b) !(a==a) || (a<b)
c) !((a!=b) && (b==b))

	Answer
a) !(a==b) && (a>b)	!(false) && (false) True && false false
b) !(a==a) || (a<b)	!(true) ||(true) false || true true
c) !((a!=b) && (b==b))	!((true) &&(true)) !(true) false

Activity 3.2C

Find the value for the following expression.
Given a = 25, b = 4;

a) a += b
b) a *= b
c) a %= b

	Answer
a) a += b	a=a + b a=25 + 4 a=29
b) a *= b	a=a * b a=25 * 4 a=100
c) a %= b	a=a % b a=25 % 4 a=5 (balance)

Activity 3.2D

Assume x, m and k is an integer variables with value x = 2 and m=6. What is the value of this expression?

a) k = m++
b) k = x* m--
c) k = m+ x * ++m

	Answer
a) k = m++	k=6 *pre increment: ignore operation after operand 'm'*
b) k = x * m--	k=6*2 k=12 *pre increment : ignore operation after operand 'm'*
c) k = m + x * ++m	k=6+2*(m=m+1); *post increment: convert ++m to m=m+1 , than continue with :* k=6+<u>2*(7)</u> k=<u>6+14</u> k=20

Exercise 3.2A

1. Find the value for the following expression.
   Given x = 10, y = 7 and z = 2

a) (x * y) % z
b) y (x + z) (x – y)
c) x / z – (x * x + y)
d) 5 (x + y + z) – x / z

Answer
a)
b)
c)
d)

Exercise 3.2B

1. Given the value a = 0, b = 6 and c = 3. Write TRUE or FALSE for the expression. Show all the steps clearly.

a) !(a==b) && (a>b)
b) ! (a == 0) && (b != 2)
c) ((a == c) && (b == c)) || (a < 1)
d) ((a + b > b) && (a == a)) || (b == a)

Answer
a)
b)
c)
d)

Exercise 3.2C

Find the value for the following expression.
Given h = 15, k = 8;

a) h -= k
b) h /= k
c) h %= k
d) h=k

Answer
a)
b)
c)
d)

Exercise 3.2D

Assume i, j and k is an integer variables with value i = 5 and j=3. What is the value of this expression?
a) k = j++
b) k = i * j--
c) k = j + i * j++
d) k = 25 / j++ - 18 % i

Answer
a)
b)
c)
d)

Exercise 3.2E

Given the following scenario. Identify the formula needed that combine operators to solve the problems.

a) You want to buy 2 different types of magazines. Given the price of the magazine and the number of magazines purchased, determine and print the total price to be paid.

b) A-ONE supermarket provides 10% discount from actual price. Find a solution to calculate the new price after discount given.

c) Find area of rectangle and user have to enter length and width of rectangle.

Answer

a)

b)

c)

Contents :

What Is Control Structure?

- Control structure is a logical structure that controls the flow of instruction to be executed by computer program.
- Control structure uses single entry and single exit meaning it has only one beginning and one ending.
- Control Structure : **SEQUENCE, SELECTION** and **LOOPING**

Sequence

- Simplest control structure.
- In this control, the program instructions are executed one by one (from top to bottom) starting from the first and ending in the last instruction.
- Every box in control structure is a process. Every process is done sequentially.

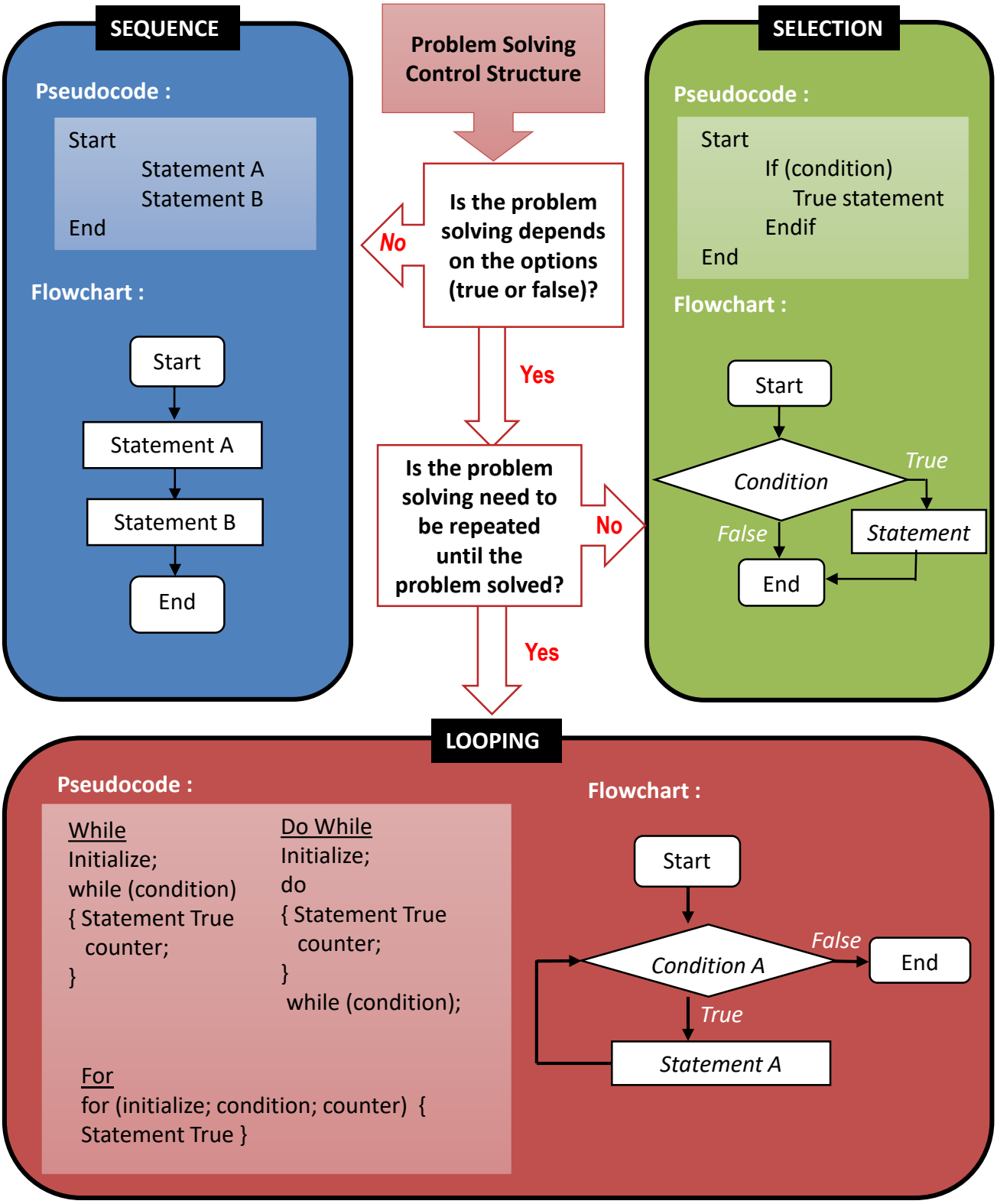
Looping

- A programming control structure that allows a series of instructions to be executed more than once.
- The similar statement is repeated several times until the conditions are fulfilled.
- Looping : **FOR, WHILE** and **DO-WHILE**

Selection

- This control structure will execute an instruction based on result of a condition either TRUE or FALSE.
- If the condition result is true, the control program will execute the instruction within the **TRUE** loop operation.
- Otherwise, it will execute the instruction within the **FALSE** loop operation.
- Selection : **IF, IF-ELSE, NESTED IF-ELSE** and **SWITCH CASE**

3.8 Control Structure Mapping



3.9 Selection Control Structure

Rule of Selection

1 Option

Types of Selection

- **If … EndIf**

Only do the instruction if condition is **TRUE**

Rule of Selection

2 Options

CONCEPT OF SELECTION

Types of Selection

- **If … Else**

Only do the instruction either condition is **TRUE** or condition is **FALSE**

Rule of Selection

Multiple options

Types of Selection

- **If … Else If**
- **Switch Case**
- **Nested If**

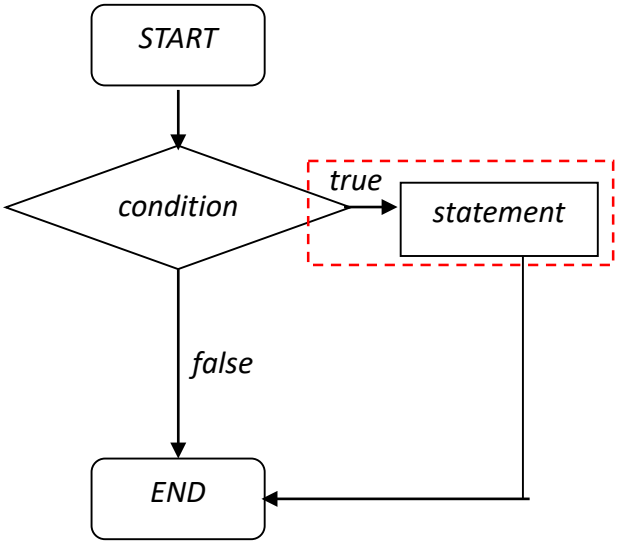
If ... EndIf

Rules:

If (condition)
Instruction (do this instruction if condition is true)
Endif

If condition is not true, no instruction will be executed

Flowchart:



Pseudocode :

```
If (condition)
        True statement

EndIf
```

Activity 3.3A : Problem solving using Selection If ...EndIf

Answer

PROBLEM & PROBLEM ANALYSIS

Workers who work on shift 3 will receive additional Bonus RM50, where basic salary is entered by workers

Input:
1. Shif
2. Basic_salary

Process:
Bonus = RM 50

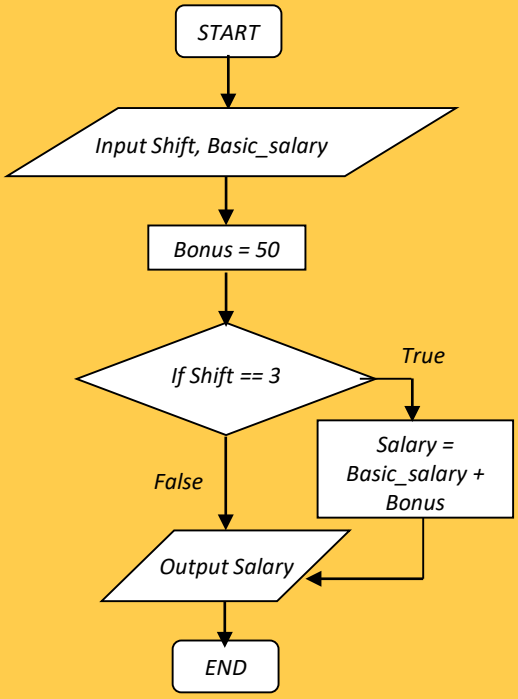
If (Shift == 3)
Salary = Bonus + Basic_salary

Output:
Salary

ALGORITHM

1.Enter Basic_salary, Shift
1.Bonus equals to RM 50
2.Check workers Shift
3.1 If Shift equals to 3
Salary= Basic_salary + Bonus
4. Display Salary

FLOWCHART

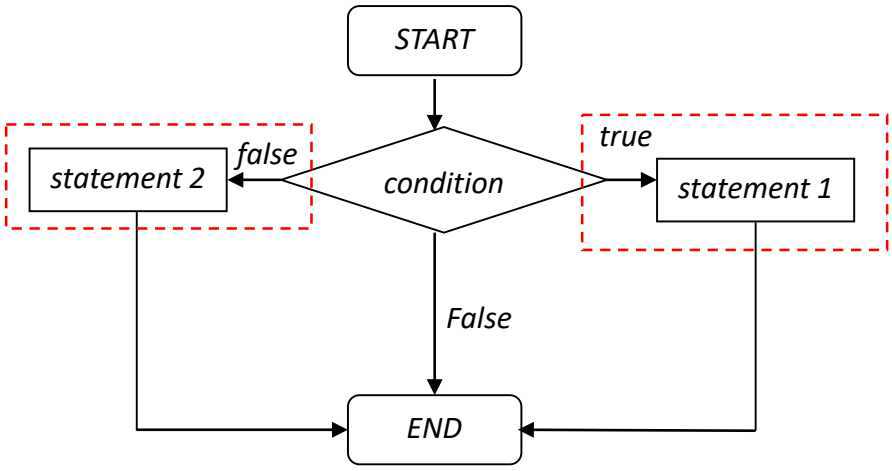


If ... Else

Pseudocode :

```
If (condition)
      statement 1

Else
      statement 2

EndIf
```

Flowchart:



Activity 3.3B : Problem solving using Selection If ...Else

Answer

PROBLEM & PROBLEM ANALYSIS

Identify whether a student is qualified to further her / his studies in any local university using his / her SPM grade equal to 1.

Input:
Grade

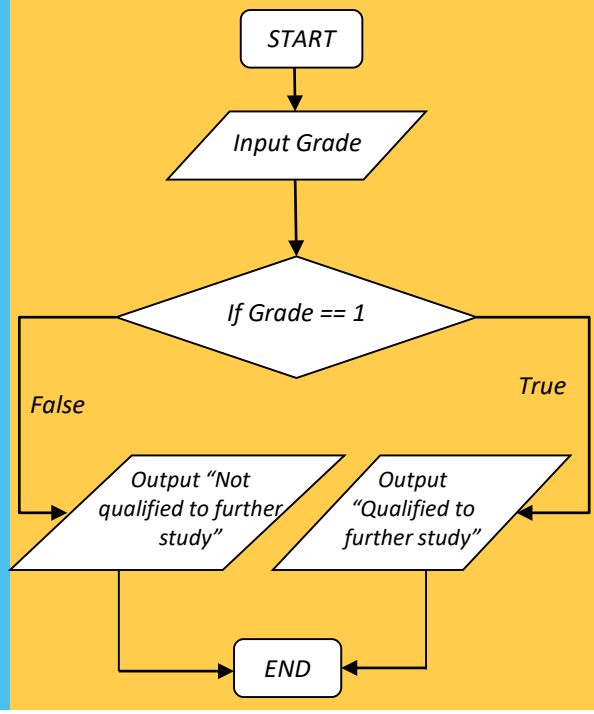
Process:
If Grade is equal to 1
Output "Qualified to further study".
If not
Output "Not qualified to further study".

Output:
Salary

PSEUDOCODE

```
START

Input Grade
If (Grade==1)
   Output "Qualified to
   further study"
Else
   Output "Not qualified
   to further study"
Endif

END
```

FLOWCHART

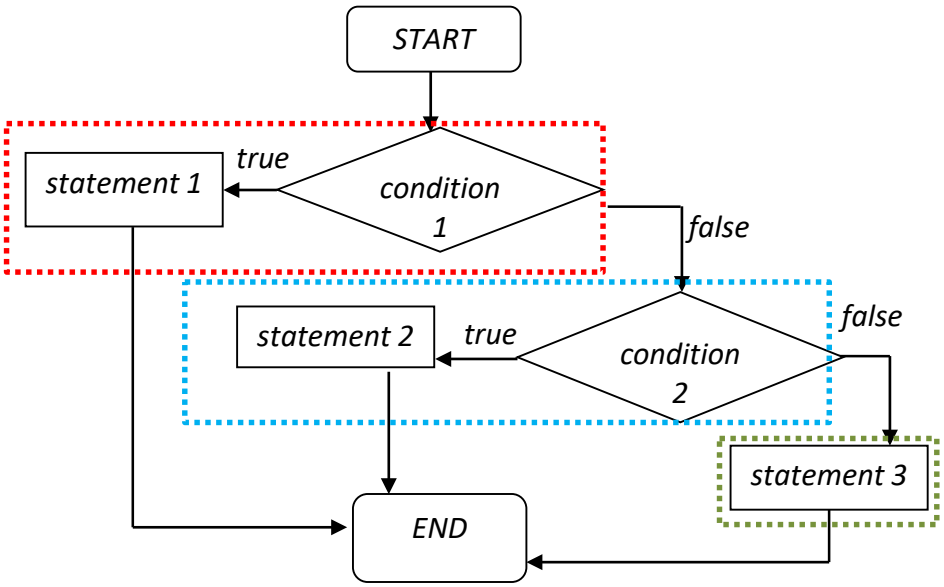


If … ElseIf

Pseudocode :

```
If (condition 1)
    statement 1
ElseIf (condition 2)
    statement 2
Else
    statement 3
Endif
```

Flowchart:



Activity 3.3C : Problem solving using Selection If …ElseIf

Answer

PROBLEM

Education status is determined based on the GPA achievement under the following scheme:

GPA	Status
3.50-4.00	Dean List
2.00-3.49	Pass
1.80-1.99	Conditional Pass
0.00-1.79	Fail

ALGORITHM

1. Enter student GPA
2. Compare student's GPA to determine his/ her education status.

2.1 If (GPA < 0.00 AND GPA > 4.00)
Output "Invalid data"
2.2 If not
2.2.1 if (GPA >=3.50 AND GPA <= 4.00)
Output "Dean List"
2.2.2 If not
2.2.2.1 If (GPA >= 2.00 AND GPA < 3.50)
Output "Pass"
2.2.2.2 If not
2.2.2.2.1 If (GPA >= 1.80 AND GPA < 2.00)
Output " Conditional Pass"
2.2.2.2.2 If not
Output "Fail"

3. Print status

PSEUDOCODE

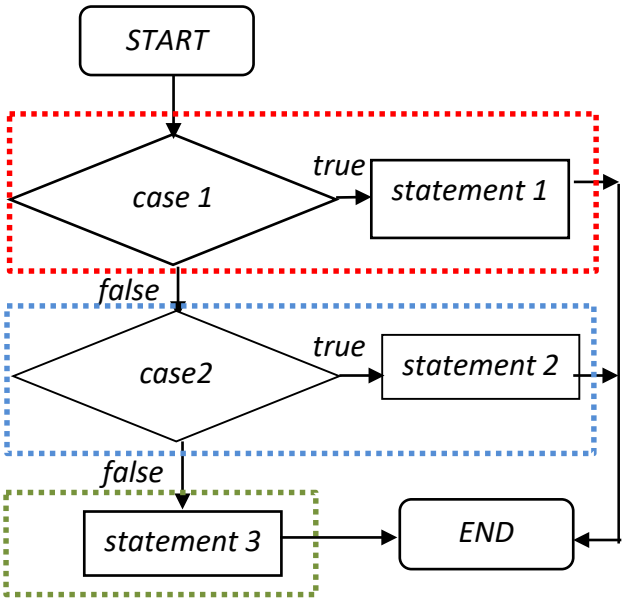
```
START
  Input GPA
    if ((GPA < 0.00) AND (GPA > 4.00))
  Output "Invalid Data"
Else if ((GPA >= 3.50) AND (GPA <= 4.00))
        Output "Dean List"
    Elseif ((GPA >=2.00) AND (GPA < 3.50))
        Output "Pass"
          Elseif ((GPA >= 1.80) AND
      (GPA < 2.00))
            Output "Conditional
Pass"
                Else
                    Output "Fail"
            Endif
    Endif
  Endif
Endif
END
```

Switch Case

Pseudocode :

```
Switch (case)
case 1 :
   statement 1
break
case 2:
   statement 2
break
default :
   statement 3
End
```

Flowchart:



Activity 3.3D : Problem solving using Selection Switch..Case

Answer

PROBLEM

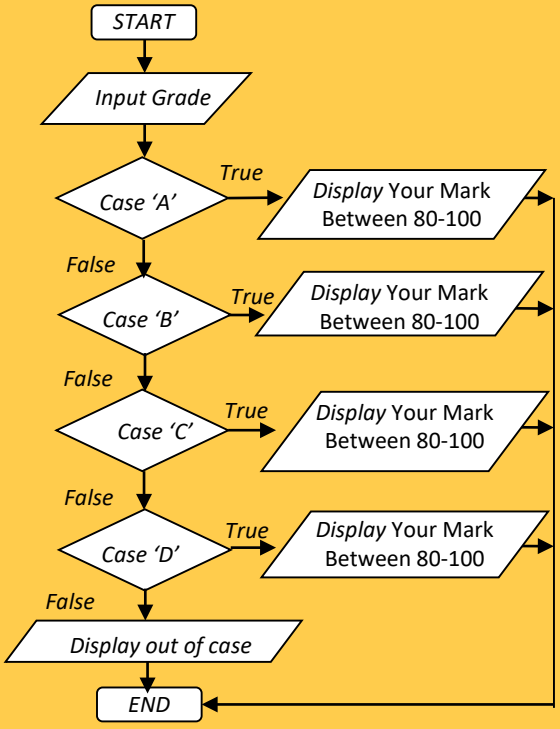
Determine a range of marks based on grade achievement based on the following table.

Grade	Mark
A	80-100
B	70-79
C	50-69
D	<50

PSEUDOCODE

Switch Grade
Case 'A':
 Display "Your Mark Between 80-100"
Case 'B':
 Display "Your Mark Between 70-79"
Case 'C':
 Display "Your Mark Between 50-69"
Case 'D':
 Display "Your Mark below than 50"
default:
 Display "Out of case"
End

FLOWCHART



Nested If

Type 1

```
If (condition1)
    If (condition2)
        If (condition3)
            Statement True
        Endif
    Endif
Endif
```

Type 2

```
If (condition1)
    If (condition2)
        If (condition3)
            Statement that will be executed if
            condition1, condition2 and condition3
            are true.
        Else
            Statement that will be executed if
            condition1, and condition2 are true but
            condition2 is false.
        Endif
    Else
        Statement that will be executed if
        condition1 is true but condition2 and
        condition3 is false.
    Endif
Else
    Statement that will be executed if condition1 is false
Endif
```

Type 3

```
If (condition1)
        Statement that will be executed if condition 1 is true
Else
        If (condition 2)
                Statement that will be executed if condition2 is
                true but condition1 is false
        Else
                If (condition3)
                        Statement that will be executed if
                        condition3 is true but condition1 and
                        condition2 are false.
                Else
                        Statement that will be executed if
                        condition1,
                        condition and condition3 are false
                Endif
        Endif
End if
```

Activity 3.3E : Problem solving using Selection Nested ...If

Answer

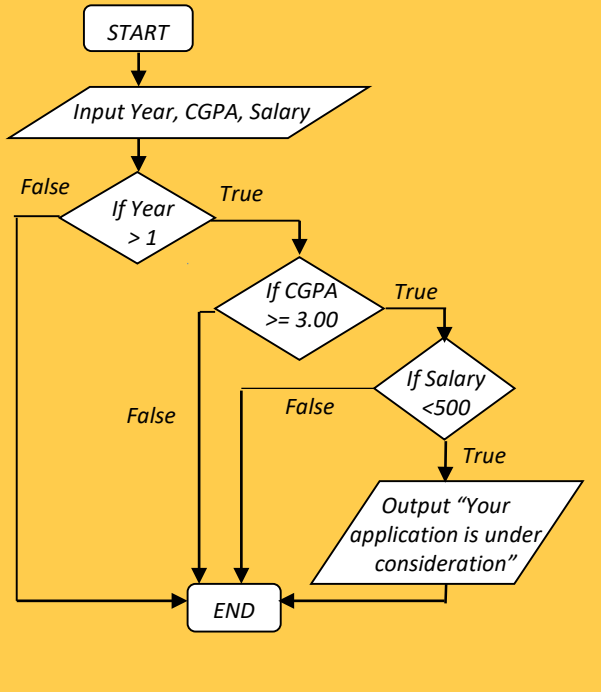
PROBLEM

To determine whether a candidate is qualified or not to get a scholarship based on his / her study years, guardian's salary and student CGPA. If the study year is more than 1, student's CGPA is not less than 3.00 and guardian's salary is below than RM500, student will be considered for a scholarship.

PSEUDOCODE

```
START

Input Year, CGPA, Salary
  If  (Year >1)
     If  (CGPA >= 3.00)
          if  (Salary < RM500)
               "Your application is
               under consideration"
          Endif
       Endif
   Endif

END
```

FLOWCHART



Exercise 3.3A

Question :

Write a pseudocode and flowchart that ask a user to enter an age. If the age is equal or more than 18 years old, print the statement “18SX Category”.

Answer

a) Pseudocode

b) Flowchart

Exercise 3.3B

Question :

Program that receives the current temperature as input. If the temperature is 80 degrees or more, output a message telling the user to go swimming, otherwise, if the temperature is 50 degrees or more, output a message to go running, otherwise stay inside. Based on problem given, write a pseudocode and flowchart.

Answer

a) Pseudocode

b) Flowchart

Exercise 3.3C

Question :

The following ID's number will determine level of user for security system. Write a pseudocode and flowchart based on table below to identify level of user for the system.

ID	Level of User
001	Super Admin
002	Admin
003	Staff
004	Member

Answer

a) Pseudocode

b) Flowchart

Exercise 3.3D

Algorithm

1. Set x = 1
2. Compare whether counter less than 26.
   If yes, go to step 3
   If no, go to step 7
3. Input name, matrix_num,mark_CA, mark_FinalTest

4. Check if the mark_CA AND mark_FinalTest is greater or equal to 40%
   4.1 If yes, Display name and matrix_num and status is "Pass"
   4.2 If no, Display name and matrix_num and status is "Fail"

5. Add counter using formula :
   x = x+1
6. Repeat step 2
7. End

Question :
Write a pseudocode and draw a flowchart for **While** loop based on the given algorithm as above.

Answer

a) Pseudocode

Answer

b) Flowchart

Exercise 3.3E

Question :

Write a pseudocode and flowchart to determine whether a candidate is qualified or not to get a scholarship based on his / her study years, guardian's salary and student CGPA. If the study year is more than 1, student's CGPA is not less than 3.00 and guardian's salary is below than RM500, student will be considered for a scholarship. If the student is not qualified the message "Not success" will be displayed.

Answer

a) Pseudocode

b) Flowchart

3.10 Looping Control Structure

FOR LOOP

```
For (initialize; condition; counter)
  {
   Statement True
  }
```

1 2 3

1 Initialize

A value to start a loop.
Must be a number.
example : x=0, i=3,
nom=1, count=20

WHILE LOOP

```
Initialize;  1
While (condition)  2
{
  Statement True
  counter  3
}
```

FLOW OF LOOPING

2 Condition

Rules for the condition:

- If the condition is true / fulfilled, the process will be performed.
- Then, the program loops back and recheck the condition, if the condition is true / fulfilled, repeat the process.
- When the condition is false, then exit the loop.

DO WHILE LOOP

```
Initialize;  1
Do
{
  Statement True
  counter  3
} While (condition);  2
```

3 Counter

Counter is to increase or decrease the initialize value.
example : i++, y-- ,
count=count + 10,
nom=nom-2

Activity 3.3A : Problem solving using For and While loop

Answer

PROBLEM & PROBLEM ANALYSIS

BMF factory have 20 paid employees with regard to basic salary, claims and overtime work. Total EPF will be deducted from basic salary by 9%. Calculate the net income earned by the employee.

Input : Basic_salary, Claim, O_time

Process :

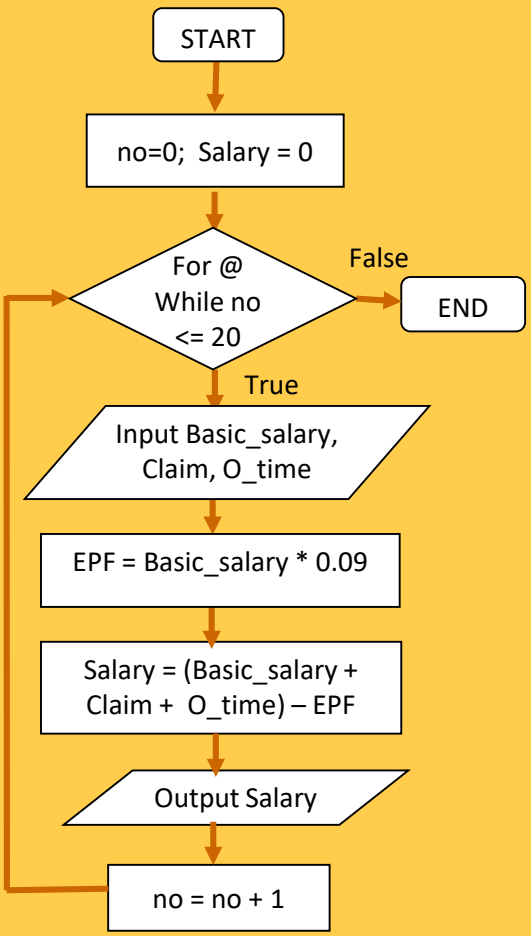
The process below will repeat until the condition to exit the loop is met.
Calculate EPF:
EPF = Basic_salary * 0.09
Calculate Salary :
Salary = Basic_salary + Claim + O_time – EPF

Output : Salary

ALGORITHM

1. Initialize Counter = 0, Salary = 0
2. Compare whether Counter is less than 20 or not
   If yes , go to step 3
   If not , go to step 9
3. Enter Basic_salary, Claim, O_time
4. Calculate EPF:
   EPF = Basic_salary * 0.09
5. Calculate Salary:
   Salary = Basic_salary + Claim + O_time – EPF
6. Display Salary
7. Add Counter using the formula:
   Counter = Counter + 1
8. Back to step 2
9. Exit loop

FLOWCHART



Activity 3.3B : Problem solving using Do While loop

Answer

PROBLEM & PROBLEM ANALYSIS

BMF factory have 20 paid employees with regard to basic salary, claims and overtime work. Total EPF will be deducted from basic salary by 9%. Calculate the net income earned by the employee.

Input : Basic_salary, Claim, O_time

Process : The process below will repeat until the condition to exit the loop is met.

Calculate EPF:
EPF = Basic_salary * 0.09

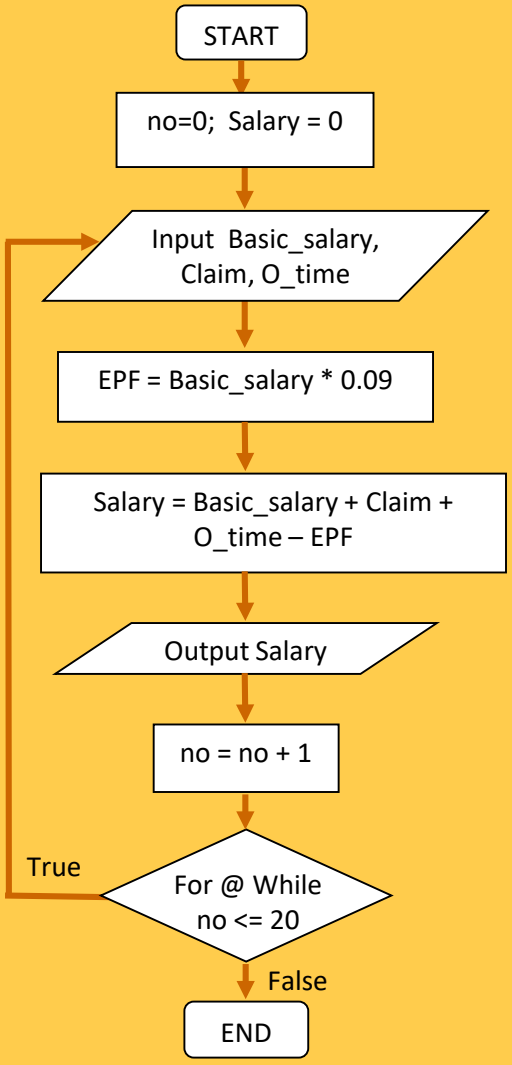
Calculate Salary :
Salary = Basic_salary + Claim + O_time – EPF

Output : Salary

ALGORITHM

1. Initialize Counter = 0, Salary = 0
2. Enter Basic_salary, Claim, O_time
3. Calculate EPF:
   EPF = Basic_salary * 0.09
4. Calculate Salary:
   Salary = Basic_salary + Claim + O_time – EPF
5. Display Salary
6. Add Counter using the formula:
   Counter = Counter + 1
7. Compare whether Counter
   is less than 20 or not
   If yes , go to step 2
   If not , go to step 9
8. Back to step 2
9. Exit loop

FLOWCHART



Exercise 3.3A

Algorithm

1. Set x = 1
2. Compare whether counter less than 26.
   If yes, go to step 3
   If no, go to step 7
3. Input name, matrix_num,mark_CA, mark_FinalTest
4. Check if the mark_CA AND mark_FinalTest is greater or equal to 40%
   4.1 If yes, Display name and matrix_num and status is "Pass"
   4.2 If no, Display name and matrix_num and status is "Fail"
5. Add counter using formula :
   x = x+1
6. Repeat step 2
7. End

Question :
Write a pseudocode and draw a flowchart for **While** loop based on the given algorithm as above.

Answer

a) Pseudocode

Answer

b) Flowchart

Exercise 3.3B

Algorithm

1. Set x = 1
2. Compare whether counter less than 26.
   If yes, go to step 3
   If no, go to step 7
3. Input name, matrix_num,mark_CA, mark_FinalTest
4. Check if the mark_CA AND mark_FinalTest is greater or equal to 40%
   4.1 If yes, Display name and matrix_num and status is "Pass"
   4.2 If no, Display name and matrix_num and status is "Fail"
5. Add counter using formula :
   x = x+1
6. Repeat step 2
7. End

Question :
Write a pseudocode and draw a flowchart for **For** loop based on the given algorithm as above.

Answer

a) Pseudocode

Answer

b) Flowchart

Exercise 3.3C

Algorithm

1. Set x = 1
2. Input name, matrix_num,mark_CA, mark_FinalTest
3. Check if the mark_CA AND mark_FinalTest is greater or equal to 40%
   3.1 If yes, Display name and matrix_num and status is "Pass"
   3.2 If no, Display name and matrix_num and status is "Fail"
4. Add counter using formula :
   x = x+1
5. Compare whether counter less than 26
   If yes, go to step 2
   If no, go to step 6
6. End

Question :
Write a pseudocode and draw a flowchart for **Do While** loop based on the given algorithm as above.

Answer

a) Pseudocode

Answer

b) Flowchart

Exercise 3.3D

Given the following scenario. Write a pseudocode and draw a flowchart for While, For and Do While loop before creating a program.

a) A class of 22 students took a quiz. The mark is between the ranges 0 - 15 for this quiz is available to you. Determine the average mark for this quiz.

b) Calculate the total salary that the company must pay to 15 employees. This calculation is based on the number of hours worked. Payment rate is RM25.50 per hour and wages will be deducted at 10% EPF.

c) Calculate the total commission received by an employee based on sales obtained for one year. Total commission is 5% of sales for each month.

Answer

a) Pseudocode

Answer

b) Flowchart

CHAPTER 4

Basic Programming Codes

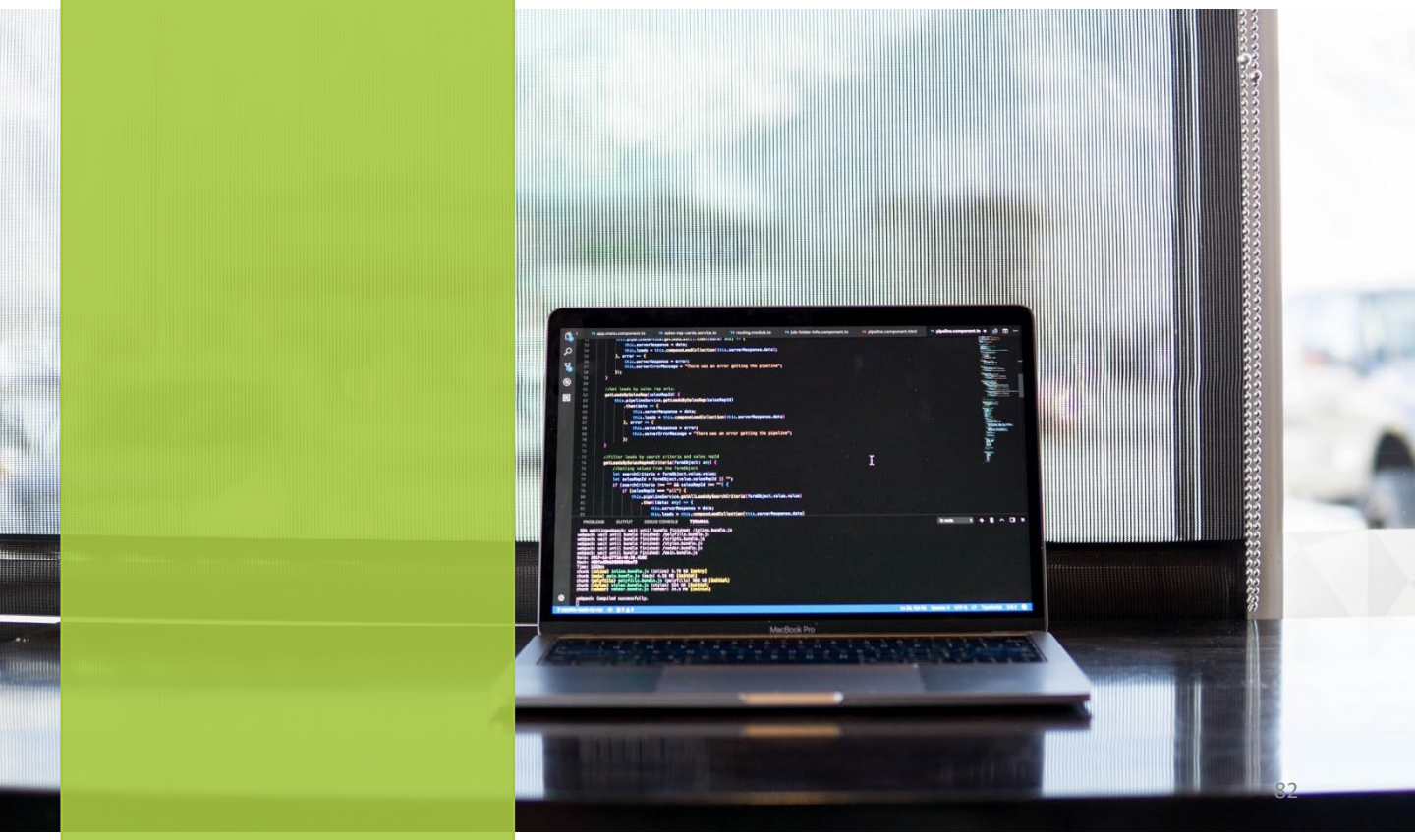


Photo by Maxwell Nelson on Unsplash

Contents :

- Element of the programming language
- Standard and best practices in programming
- Steps in creating a C++ program
- Basic program concepts
- Analyse program to identify input, process and output
- Construct a pseudo code based on given programming code
- Construct a flowchart based on given programming code

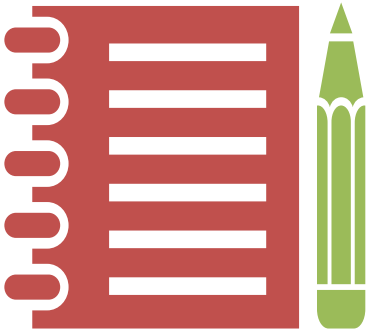
Elements of the Programming language

- The element used to write a programming code.
- The component that build a programming codes.

Process

- The main part and engine of the program development.
- The engine of programming code

QUICK FACTS



Input

- User input data by keyboard or text file.
- A **MUST** element in a programming code for interactive programming.

Output

- The outcomes from the programming process.
- What the user will acquire from the programming code.

4.1 Element of the Programming Language : a quick view

1. Comment

- to document and understand the program.
- // - single line
- /* */ - multiple paragraph

2. Preprocessor directive

- Starts with #.
- Used to include header files.

3. Standard header file

- Stores functions that are related to a particular application.
- To reduce programming time.

4. Main function

- A block code that runs a task.
- A **MUST** in a program.

5. Reserved word

- A special word reserved by a programming language or by a program

6. Identifiers

- The name of the variable, constant and function in program.

7. Special symbol

- To represent the basic arithmetic operations
- e.g. + , - , * , / , %

8. Punctuation

- Every statement must end with a semicolon ' ; '.

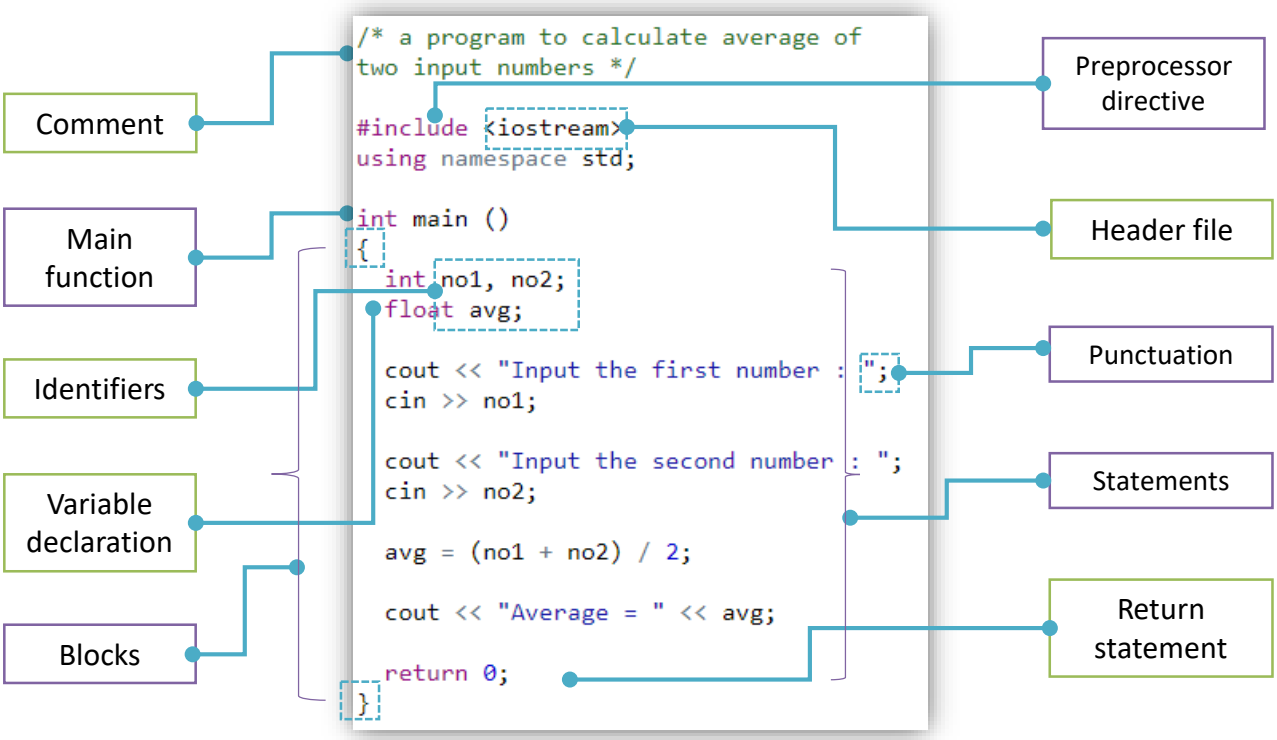
9. Statements

- An instruction including input, output, process, variable declaration.

10. Blocks

- Opening braces ' { ' and closing braces ' } ' of the function.
- { … }

4.2 The elements in Practice



Activity 4.1

1. Identify the basic elements of the programming code below.

```
// a program to display a welcome notes

#include <iostream>
using namespace std;

int main()
{
    cout<<"Welcome to C++ Programming Language!";
    return 0;
}
```

Answer

Element	Programming Code
Comment	`// a program to display a welcome notes`
Preprocessor directive	`#include`
Header file	`iostream`
Main Function	`int main()`
statement	`cout<<"Welcome to C++ Programming Language!";`
Punctuation	;
Return statement	`return 0;`

Exercise 4.1

```
#include <iostream>
using namespace std;

void main(){
    float pi = 3.142;
    float radius, area;

    cout<<"Input the radius: ";
    cin>> radius;

    area = pi * radius * radius;

    cout<<"Area = "<<area;
    system ("pause");
}
```

Question :

Based on the given program code, write the program code for element of programming

a. Main function
b. Block
c. Reserved word
d. Header file
e. Identifier

Answer

a.

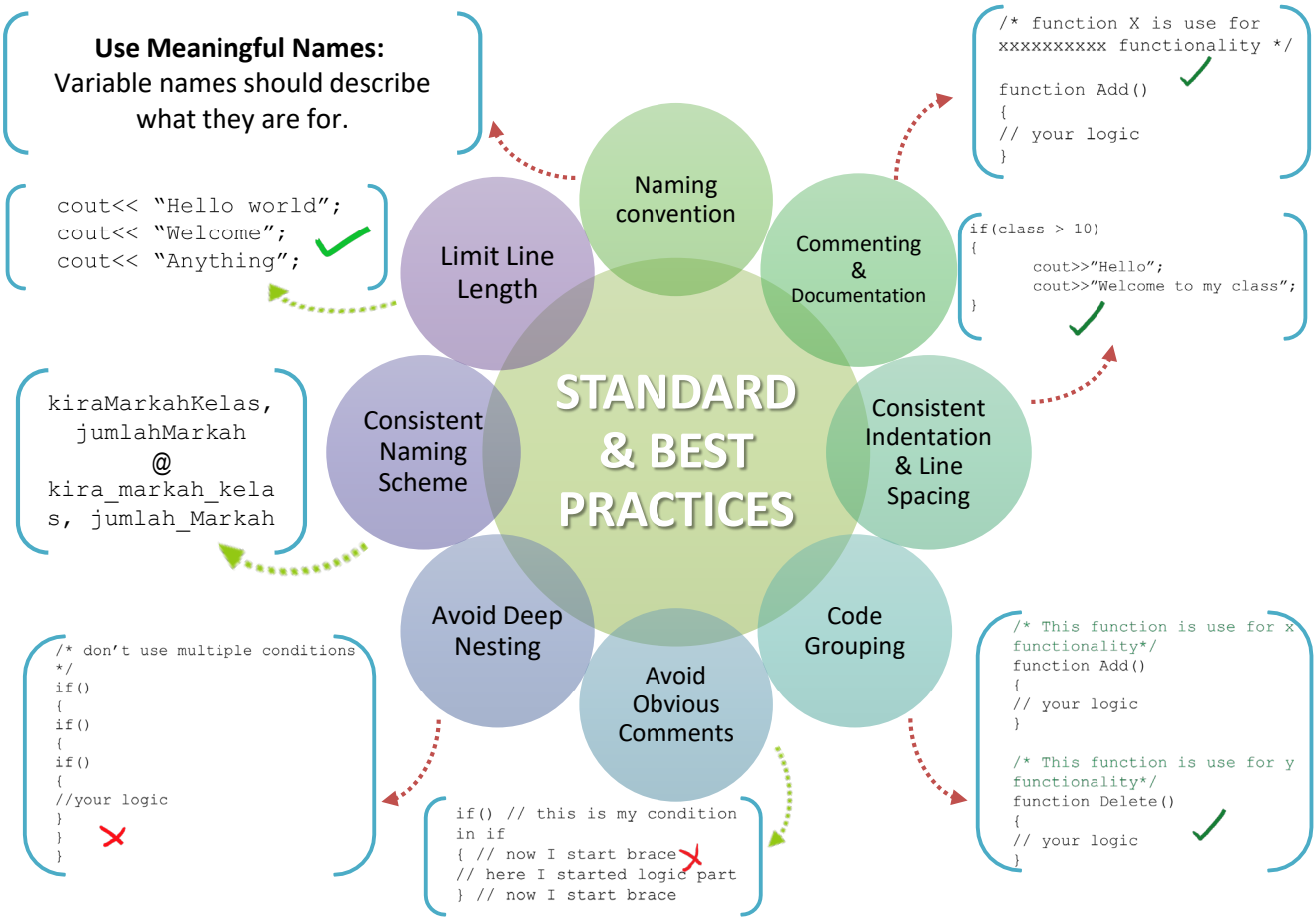
b.

c.

d.

e.

4.3 Standards & Best Practices in Programming



How it Works?

Refer to the given examples to get the better understanding for each of the standard and best practices.

4.4 Steps in creating a C++ program

```
#include <iostream>
using namespace std;

void main(){
    float pi = 3.142;
    float radius, area;

    cout<<"Input the radius: ";
    cin>> radius;

    area = pi * radius * radius;

    cout<<"Area = "<<area;
    system ("pause");
}
```

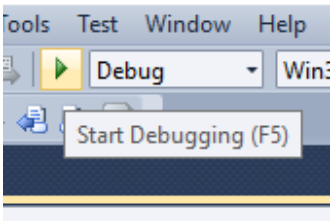
What you'll do in C++

1. Type the source code
2. Compile the program
3. Run the compile code to get the output

Source

1

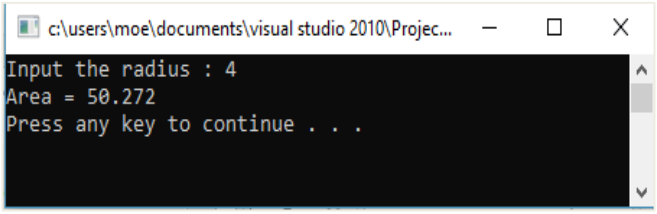
Type your source code
File extension .cpp



Compiler

2

Compile the program to check the error.



Output

3

The output generated from the compiled program based on user input to the program

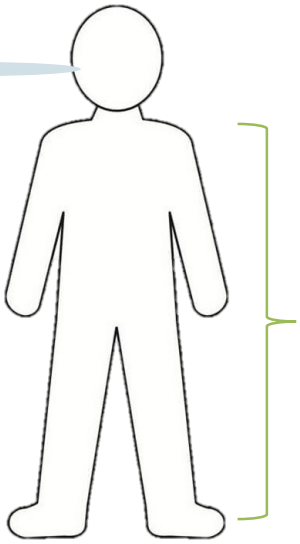
4.5 Basic Program Concepts

HEAD

- The first thing load in the program.
- **Element :** Preprocessor Directive, header file.

Code Example :

```
#include <iostream>
using namespace std;
```



BODY

- The main part of the programming.
- The heart of the program.
- **Main Element :** Main function

Code Example :

```
int main ()
{
  int no1, no2;
  float avg;

  cout << "Input the first number : ";
  cin >> no1;

  cout << "Input the second number : ";
  cin >> no2;

  avg = (no1 + no2) / 2;

  cout << "Average = " << avg;

  return 0;
}
```

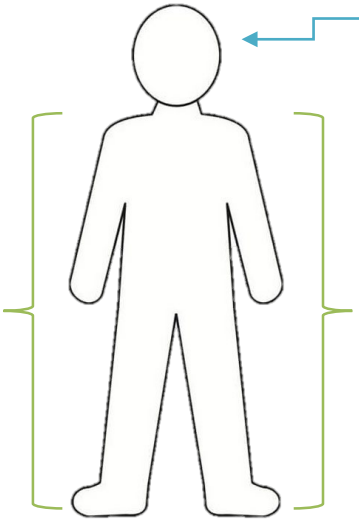
4.6 Pseudocode to program codes

Steps to be follow:

1. Write the header part.
2. Then, complete the code for main function.

Head

```
#include <iostream>
using namespace std;
```



```
START

    Pi = 3.142
    Input radius
    area = Pi * radius *radius
    Output area
END
```

Pseudocode

```
void main(){
    float pi = 3.142;
    float radius, area;

    cout<<"Input the radius: ";
    cin>> radius;

    area = pi * radius * radius;

    cout<<"Area = "<<area;
    system ("pause");
}
```

Main function

4.6 Pseudocode to program codes

Reminder :

1. All the **variables** used in the program **MUST** be declared.

Main function

Head

```
START

    Pi = 3.142
    Input radius
    area = Pi * radius *radius
    Display area

END
```

Pseudocode

pseudocode to calculate area of circle

```
#include <iostream>
using namespace std;

void main(){
    float pi = 3.142;
    float radius, area;

    cout<<"Input the radius: ";
    cin>> radius;

    area = pi * radius * radius;

    cout<<"Area = "<<area;
    system ("pause");
}
```

Variable declaration

The complete program code

A C++ code to calculate area of circle.

How it Works?

The arrow line from the pseudocode pointing to the matching statement for C++ code.

4.7 Analyze a program to identify input, process and output

```
#include <iostream>
using namespace std;

void main(){
    float pi = 3.142;
    float radius, area;

    cout<<"Input the radius: ";
    cin>> radius;

    area = pi * radius * radius;

    cout<<"Area = "<<area;
    system ("pause");
}
```

Input

- The data used in the program for process purposes.
- The data might be response by user using input device or by read file.

Process

- The operations like arithmetic and logical operations.

Output

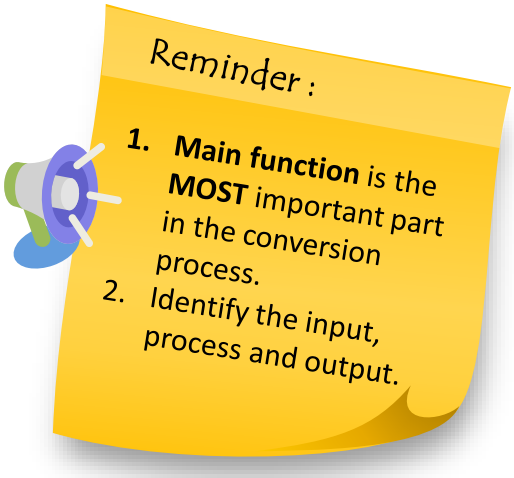
- The results from the process involved.
- The information that end user will see / received from the program.

Notes :

cin>> - to input data
cout<< - to display information

BOTH are the functions from header file **iostream.**

4.8 Construct a pseudo code based on given programming code




```cpp
/* a program to calculate average of
two input numbers */

#include <iostream>
using namespace std;

int main ()
{
  int no1, no2;
  float avg;

  cout << "Input the first number : ";
  cin >> no1;

  cout << "Input the second number : ";
  cin >> no2;

  avg = (no1 + no2) / 2;

  cout << "Average = " << avg;

  return 0;
}
```

A C++ code to calculate average of two input numbers

```
START
    Read int no1, no2
    Input the first number : no1
    Input the second number : no2
    avg = (no1 + no2) / 2
    Print avg
END
```

Pseudo Code

Pseudo code to calculate average of two input numbers

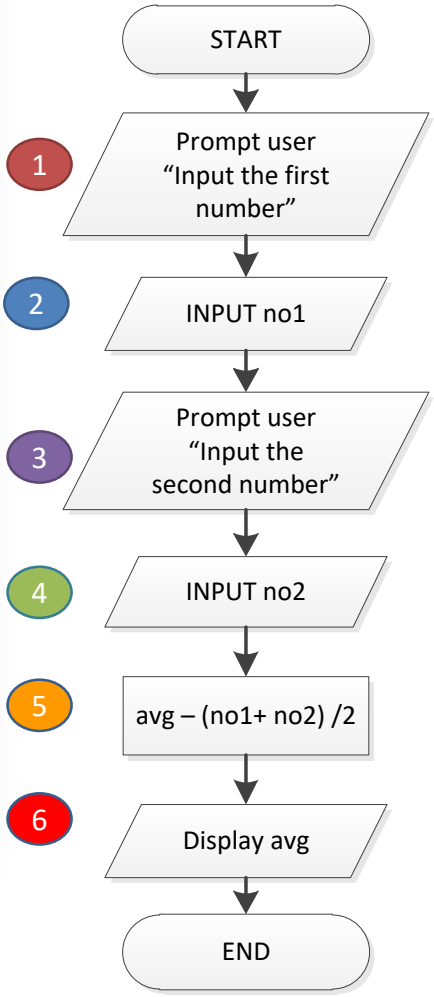
How it Works?

The arrow line from the C++ code pointing to the matching statement for pseudo code.

4.9 Construct a flowchart based on given programming code.

```
/* a program to calculate average of
two input numbers */

#include <iostream>
using namespace std;

int main ()
{
  int no1, no2;
  float avg;

1 cout << "Input the first number : ";
2 cin >> no1;

3 cout << "Input the second number : ";
4 cin >> no2;

5 avg = (no1 + no2) / 2;

6 cout << "Average = " << avg;

  return 0;
}
```

Pseudo code to calculate average of two input numbers



Flowchart to calculate average of two input numbers

How it works?

Mapping each of the number from the pseudo code to the number from the flowchart. Each of the number show the conversion from the command in pseudo code to the symbol in flowchart.

Activity 4.2

```
/* a program to calculate average of
two input numbers */

#include <iostream>
using namespace std;

int main ()
{
  int no1, no2;
  float avg;

  cout << "Input the first number : ";
  cin >> no1;

  cout << "Input the second number : ";
  cin >> no2;

  avg = (no1 + no2) / 2;

  cout << "Average = " << avg;

  return 0;
}
```

Question :

1. Create a file average.cpp
2. Write the program code
3. Compile the program
4. Generate the output from the program
5. Write the output of the program
6. Identify the input, process and output of the program.

Answer

5. **Output :**

```
Input the first number: 5
Input the second number: 45
Average = 25
```

6. **Input** : no1, no2
   Process : avg = (no1 + no2) /2
   Output : avg

Exercise 4.2

```
cout << "Input the percent of discount: ";
disc_price = price - discount;
return 0;
cin >> price;
cin >> percent;
float discount, discounted_price;
int price, percent;
int main ()
#include <iostream>
using namespace std;
/*Calculate the the price
of item after discount */
{
}
discount = price * percent / 100 ;
cout << "Input the price of item: RM";
cout << "The new price = RM"<< disc_price ;
```

Output :

```
Input the price of item: RM45
Input the percent of discount: 15
The new price = RM39
```

Question : Arrange the scrambled up snippet code to make a working C++ program that produce the output listed.

Answer

Exercise 4.3

```
/*Calculate salary for
worker based on hour worked */

#include <iostream>
using namespace std;

int main ()
{
  int hour;
  float rate, salary;
  cout << "Input the hour worked: ";
  cin >> hour;
  cout << "Input rate per hour: RM  ";
  cin >> rate;
  salary = hour * rate;
  cout << "The wages = RM"<< salary ;
  return 0;
}
```

Question :
Write a pseudocode and draw a flowchart for based on the above program code.

Answer

a) Pseudocode

b) Flowchart

Exercise 4.4

Miss Amelia are given you a task to help Finance department in completing payroll system. There will be 10% increment for gross pay salary for each staff. Create a program code to calculate the new salary for each staff including the allowance.

Question :
Follow the process below in completing your codes.

Task 1: You must identify the input, process and output from the given task

Task 2: Construct a flowchart of the given task based on **Task 1**.

Task 3: Construct a pseudocode of the given task based on **Task 2**.

Task 4: Write a complete code of the given task based on **Task 3**.

Answer

Task 1 :

INPUT :

PROCESS :

OUTPUT:

Task 2 : Flowchart

Answer

Task 3 :

Task 4 :

REFERENCES

Kenneth. C. L. &. Kenneth. A. L. (2020). *Programming Languages: Principles and Practices (Advanced Topics) 3rd Edition*. Cengage Learning.

Somashekara. M.T., Guru. D. S., &. Manjunatha. K. S. (2018). *Problem Solving with C (2nd Edition)*. India. PHI Learning.

Wan Mohammad, W.A, Mohd Mydin, A. and Ishak, S. (2015). *Introduction to C++ Programming*. Shah Alam. Oxford University Press.

Yew, K.H. (2018). *Fundamentals of Programming*. Shah Alam. Oxford Fajar.

Unsplash. (n.d.). *Beautiful free images & pictures*. Unsplash. Retrieved November 18, 2021, from https://unsplash.com/.

Terbitan

POLITEKNIK
MALAYSIA
KUALA TERENGGANU

e ISBN 978-967-2240-20-4

9 7 8 9 6 7 2 2 4 0 2 0 4