**Polytechnic University of the Philippines**

College of Computer and Information Sciences

*Department of Computer Science*

Sta. Mesa, Manila

In Partial Fulfillment of the

Requirements for the Subject

COSC 3053: Principles of Programming Languages

PUTOK Programming Language

Calvadores, Angelica

Enrique, Mariella

Loto, Ramzel Renz

Roman, Rose Anne

BSCS 2-1N

Prof. Elias Austria

December 18, 2015

# PROGRAMMING LANGUAGE

PUTOK PROGRAMMING LANGUAGE

# DESCRIPTION

PPL (Putok Programming Language) is a general-purpose, imperative computer programming language based from C programming language but to be more user-friendly. Also, it is to have combined features of most languages.

# 11 SYNTACTIC ELEMENTS

# Character Set

* Alphabets
  + Uppercase: A B C …. X Y Z
  + Lowercase: a b c …. x y z
* Numerals
  + 0 1 2 3 4 5 6 7 8 9
* Special Characters
  + ., ,, /, >, <, =, !, @, #, $, %, ^, &, \*, (, ), \_, -, [, ], {, }, |, +

1. Identifiers

An identifier is a [name](https://en.wikipedia.org/wiki/Name) that identifies (that is, labels the [identity](https://en.wikipedia.org/wiki/Identity_(philosophy)) of) either a unique object or a unique class of objects.

Rules for naming identifiers:

* No identifiers should be named the same as any reserved words.
* Identifiers should not begin with a number, and contain any spaces and special symbols.
* Identifiers can contain a number.

1. Operators

* Arithmetic Operators

OPERATOR

* + \* Multiplication 8 \* 2 = 16 or 8 mult 2 = 16
  + / Division 8 / 2 = 4 or 8 div 2 = 4
  + + Addition 8 + 2 = 10 or 8 add 2 = 10
  + – Subtraction 8 – 2 = 6 or 8 minus 2 = 6
  + % Modulus 8 % 2 = or 8 mod 2 =
* Relational Operators
  + &&
    - Logical AND - produces the value 1 if both operands have non-zero values. If either operand is equal to 0, the result is 0. If the first operand of a logical-AND operation is equal to 0, the second operand is not evaluated.
    - Example: (
  + ||
    - Logical OR - operator performs an inclusive-OR operation on its operands. The result is 0 if both operands have 0 values. If either operand has a nonzero value, the result is 1. If the first operand of a logical-OR operation has a nonzero value, the second operand is not evaluated.
  + ! –
    - Logical NOT - the operator returns the opposite Boolean value of evaluating its operand.
* Relational Operators
  + == equal to if chiffon == mousse
  + > greater than if chiffon > mousse
  + >= greater than or equal to if chiffon >= mousse
  + < less than if chiffon < mousse
  + <= less than or equal to if chiffon <= mousse
  + != not equal to if chiffon != mousse
* Assignment operator
  + = takes the value of bread = “ Half roll “
* Increment….
  + ++
  + --

1. Keywords
   * Decision control structure
     + If, else, if-else, if-else-if, do, while, do-while, for
2. Reserved Word
   * int, let (letter), word (string), true, false, dec (double/float)
3. Comments
   * Enclosed in pound sign (#)

e.g., # … sample comment … # or

# sample comment …

# sample comment …

#

1. Blank / Spaces

/BR/ - refers to a break line

/TAB/ - refers to a tabbed space

1. Delimiters and brackets
   * Open and close parentheses () and {} ( curly braces ) are used for grouping statements.
   * Statements ends with a semicolon.
2. Free and Fixed-Field Formats
   * Putok Programming Language is a free-field format because it does not require any proper formatting for the program to execute without any errors.
   * All statements are always terminated by a semicolon.
3. Expressions
   * Arithmetic Expressions

+ or add Addition a + b

- or minus Subtraction a – b

\* or mult Multiplication a \* b

/ or div Division a / b

% or mod Modulus a % b

++ Increment a++, ++a

-- Decrement a--, --a

* + Relational Expressions

== Equal to a == b

> Greater than a > b

>= Greater than equal a >= b

< Less than a < b

<= Less than equal a <= b

!= Not equal a != b

* + Logical Expressions

&& Logical AND a && b

|| Logical OR a || b

! Logical NOT !a, !b

* + Boolean Expressions

TRUE True

FALSE False

1. Statements
   * Comment

Line comment - #comment

Line comments can be used for debugging, explaining a code, or putting an instruction for other programmers.

* Input / Output Statements

Input: bake( “%format\_specifier”, identifier );

Output: chill( “ “ );

* Declaration Statements

<data type> variable name;

Example: int chiffon;

* Assignment Statements

variable\_name = value;

e.g. chiffon = 500;

variable\_name = variable\_name;

e.g., chiffon = mousse;

variable\_name = expression;

e.g., chiffon = a + b;

* Selection Statements

Two – way condition:

If (expr) else {(statement)}

Nested two-way conditions:

If (expr) else {(statements)}

If (expr) else {(statements)}

If (expr)

If only:

If (expr) else {(statements)}

If (expr) else {(statements)}

* Iteration Statements

for(expr;expr;expr)

{(statements}

1. GRAMMAR RULE, DERIVATIONS, PARSE TREE