ALevel CS C14 Programming (1) Java

Declare of variables:

pseudocode:
DECLARE <identifier> : <dataType>
Example
DECLARE Number1 : INTEGER // this declares Number1 to store a whole number
DECLARE YourName : STRING // this declares YourName to store a
// sequence of characters
DECLARE N1, N2, N3 : INTEGER // declares 3 integer variables
DECLARE Name1, Name2 : STRING // declares 2 string variables
Java:
int number1;
String yourName;
int n1, n2, n3;
String name1, name2;

Declare and assignment of constants:

pseudocode:
CONSTANT <identifier> = <value>
Example
CONSTANT Pi = 3.14
Java:
static final double PI = 3.14;

Assignment of variables:

pseudocode:
<identifier> ← <expression>
Example
A ← 34
B ← B + 1
Java:
A = 34;
B = B + 1;

Description	pseudocode	java
While signed numbers	INTEGER	int(4 bytes)
signed numbers with a decimal point	REAL	float (4 bytes) double(8
A single character	CHAT(single ' quotation)	char(2 bytes - Unicode)
A sequence of characters(a String)	STRING(doubl " quotation)	String (2 bytes per character)
Logical values	BOOLEAN	Boolean true false
Date value	DATE	import java.util.Date

operation	pseudocode	java
Addition	+	+
Substractior	-	-
multiplicatio	*	*
Division	/	/
Exponent	^	/
Integer division	DIV	/
Modulus	MOD	%

Operation	pseudocode	java
equal	=	==
not equal	<>	!=
greater than	>	>
less than	<	<
greater than or equal to	>=	>=
less than or equal to	<=	<=

Operation	pseudocode	java
AND (logical conjunction)	AND	&&
OR (logical inclusion)	OR	II
NOT (logical negation)	NOT	!

Outputing information:

pseudocode:
OUTPUT <string>
OUTPUT <identifier(s)>
Examples:
OUTPUT "Hello ", YourName, ". Your number is ", Number1 // newline
OUTPUT "Hello " // no new line
Java:
// syntax
System.out.print(<printlist>);
System.out.println(<printlist>);
//Examples
System.out.println("Hello " + yourName + ". Your number is " + number1);
System.out.print("Hello");

Get input from the user:

pseudocode:
INPUT "Enter a number: " A
Java:
import java.util.Scanner;
Scanner console = new Scanner(System.in);
System.out.print("Enter a number: ");
a = console.next();

Comments:

Java:
// this is a comment
// this is another comment
/* this is a multi-line comment */

Selection

pseudocode:
IF <Boolean expression>
  THEN
    <statement(s)>
ENDIF
IF <Boolean expression>
  THEN
    <statement(s)>
  ELSE
    <statement(s)>
ENDIF
# Examples
IF x < 0
  THEN
    OUTPUT "Negative"
ENDIF

IF x < 0
  THEN
    OUTPUT "Negative"
  ELSE
    OUTPUT "Positive"
ENDIF

Java:
// Syntax
if (<Boolean expression>)
  <statement>;

if (<Boolean expression>)
  <statement>;
else
  <statement>;
// Examples
if (x < 0)
  System.out.println("Negative");

if (x < 0)
  System.out.println("Negative");
else
  System.out.println("Positive");

Truncating numbers

pseudocode:
INT(x : REAL) RETURNS INTEGER
STRING _ TO _ NUM(x : STRING) RETURNS REAL
Java:
Integer.valueOf(S)
Float.valueOf(x)

CASE condition:

pseudocode:
CASE OF <expression>
  <value1> : <statement(s)>
  <value2>,<value3> : <statement(s)>
  <value4> TO <value5> : <statement(s)>
  .
  .
  OTHERWISE <statement(s)>
ENDCASE
Example:
CASE OF Grade
  "A" : OUTPUT "Top grade"
  "F", "U" : OUTPUT "Fail"
  "B".."E" : OUTPUT "Pass"
OTHERWISE OUTPUT "Invalid grade"
ENDCASE
Java:
switch (grade) {
  case 'A':
    System.out.println("Top Grade");
    break;
  case 'F': case 'U':
    System.out.println("Fail");
    break;
  case 'B': case 'C': case 'D': case 'E':
    System.out.println("Pass");
    break;
  default:
    System.out.println("Invalid grade");
}

Post-condition loops

pseudocode:
REPEAT
  <statement(s)>
UNTIL <condition>
Example:
REPEAT
  INPUT "Enter Y or N: " Answer
UNTIL Answer = "Y"
Java:
do
{
  <statement(s)>
} while <condition>;

//Examples
do
{
  System.out.print("Enter Y or N: ");
  answer = console.next();
} while (!(answer.equals("Y")));

Count-controller(FOR) loops

pseudocode:
FOR <control variable> ← s TO e STEP i // STEP is optional
  <statement(s)>
NEXT <control variable>
Example:
FOR x ← 1 TO 5
  OUTPUT x
NEXT x

FOR x = 2 TO 14 STEP 3
  OUTPUT x
NEXT x
FOR x = 5 TO 1 STEP -1
  OUTPUT x
NEXT x
Java:
for (int x = 1; x < 6; x++)
{
  System.out.print(x);
}

for (int x = 2; x < 15; x = x + 3)
{
  System.out.print(x + " ");
}
for (int x = 5; x > 0; x--)
{
  System.out.print(x + " ");
}
char[] letter = {'a', 'b', 'c'};
for (char x : letter)
{
  System.out.print(x);
}

Random number generator

Java:
import java.util.Random;
Random randomNumber = new Random();
int x = randomNumber.nextInt(6) + 1;

Pre-condition loops

pseudocode:
WHILE <condition> DO
  <statement(s)>
ENDWHILE
Example:
Answer ← ""
WHILE Answer <> "Y" DO
  INPUT "Enter Y or N: " Answer
ENDWHILE
Java:
while (<condition>)
{
  <statement(s)>;
}

//Examples
String answer = "";
while(answer.equals("Y") == false)
{
  System.out.print("Enter Y or N: ");
  answer = console.next();
}

Procedure

pseudocode:
PROCEDURE <procedureIdentifier> // this is the procedure header
  <statement(s)> // these statements are the procedure body
ENDPROCEDURE
CALL <procedureIdentifier>
Examples
PROCEDURE InputOddNumber
  REPEAT
    INPUT "Enter an odd number: " Number
  UNTIL Number MOD 2 = 1
  OUTPUT "Valid number entered"
ENDPROCEDURE
CALL InputOddNumber
Java:
void <identifier> ()
{
  <statement(s)>;
}

Functions

pseudocode:
FUNCTION <functionIdentifier> RETURNS <dataType> // function header
  <statement(s)> // function body
  RETURN <value>
ENDFUNCTION
Examples
FUNCTION InputOddNumber RETURNS INTEGER
  REPEAT
    INPUT "Enter an odd number: "
Number
  UNTIL Number MOD 2 = 1
  OUTPUT "Valid number entered"
  RETURN Number
ENDFUNCTION

ALevel CS C14 Programming (2) Java

Description	pseudocode	java
Access a single character using its position P in a string ThisString	ThisString[P] Counts from 1	ThisString.charAt(P) Counts from 0
Return the character whose ASCII value is	chr(i)	(char) i;
Returns the ASCII value of character ch	ord(ch)	(int) ch;
Returns the integer value representing the length of String S	LENGTH(S : STRING) RETURNS INTEGER	S.length();
Returns leftmost L characters from S	LEFT(S : STRING, L : INTEGER) RETURNS STRING	S.subString(0, L)
Returns rightmost L characters from S	RIGHT(S : STRING, L : INTEGER) RETURNS STRING	S.subString(S.length() - L)
Returns a string of length L starting at position P from S	MID(S : STRING, P : INTEGER, L : INTEGER) RETURNS STRING	S.subString(P, P + L)
Returns the character value representing the lower case equivalent of Ch	LCASE(Ch : CHAR) RETURNS CHAR	Character.toLowerCase(ch)
Returns the character value representing the upper case equivalent of Ch	UCASE(Ch : CHAR) RETURNS CHAR	Character.toUpperCase(ch)
Returns a string formed by converting all alphabetic characters of S to upper case	TO_UPPER(S : STRING) RETURNS STRING	S.toUpperCase()
Returns a string formed by converting all alphabetic characters of S to lower case	TO_LOWER(S : STRING) RETURNS STRING	S.toLowerCase()
Concatenate(join) two strings	S1 & S2	s = S1 + S2;

Passing parameters to procedure

pseudocode:

```
# procedure header
PROCEDURE <ProcedureIdentifier> (<parameterList>)
# parameter
BYREF <identifier1> : <dataType>
BYVALUE <identifier2> : <dataType>
```

Passing parametrs by value

```
PROCEDURE OutputSymbols(BYVALUE NumberOfSymbols : INTEGER, Symbol
: CHAR)
   DECLARE Count : INTEGER
   FOR Count ← 1 TO NumberOfSymbols
      OUTPUT Symbol // without moving to next line
   NEXT Count
   OUTPUT NewLine
ENDPROCEDURE
```

Passing parameters by reference

```
PROCEDURE AdjustValuesForNextRow(BYREF Spaces : INTEGER, Symbols :
INTEGER)
   Spaces ← Spaces - 1
   Symbols ← Symbols + 2
ENDPROCEDURE
CALL AdjustValuesForNextRow(NumberOfSpaces, NumberOfSymbols)
```

Text files

pseudocode:

```
# Writing to a text file
OPENFILE <filename> FOR WRITE // open the file for writing
WRITEFILE <filename>, <stringValue> // write a line of text to the file
CLOSEFILE <filename> // close file

# Reading from a text file
OPENFILE <filename> FOR READ // open file for reading
READFILE <filename>, <stringVariable> // read a line of text from the file
CLOSEFILE <filename>  // close file

# Appending to a text file
OPENFILE <filename> FOR APPEND // open file for append
WRITEFILE <filename>, <stringValue> // write a line of text to the file
CLOSEFILE <filename> // close file

# The end-of-file (EOF) marker
OPENFILE "Test.txt" FOR READ
WHILE NOT EOF("Test.txt") DO
   READFILE "Test.txt", TextString
   OUTPUT TextString
ENDWHILE
CLOSEFILE "Test.txt"
```

Passing parameters to subroutings

pseudocode:

```
# function header

FUNCTION <functionIdentifier> (<parameterList>) RETURNS <dataType>

FUNCTION SumRange(FirstValue : INTEGER, LastValue : INTEGER) RETURNS
INTEGER
   DECLARE Sum, ThisValue : INTEGER
   Sum ← 0
   FOR ThisValue ← FirstValue TO LastValue
      Sum ← Sum + ThisValue
   NEXT ThisValue
   RETURN Sum
ENDFUNCTION
```

Arrays

pseudocode:

```
DECLARE <arrayIdentifier> : ARRAY[<lowerBound>:<upperBound>] OF <dataType>
DECLARE List1 : ARRAY[1:3] OF STRING // 3 elements in this list
DECLARE List2 : ARRAY[0:5] OF INTEGER // 6 elements in this list
DECLARE List3 : ARRAY[1:100] OF INTEGER // 100 elements in this list
DECLARE List4 : ARRAY[0:25] OF STRING // 26 elements in this list
# accessing 1D arrays
<arrayIdentifier>[x]
# examples
NList[25] = 0 // set 25th element to zero AList[3] = "D" // set 3rd element to letter D
# creating 2D arrays
DECLARE <identifier> : ARRAY[<lBound1>:<uBound1>, <lBound2>:<uBound2>] OF
<dataType>
# examples
DECLARE Board : ARRAY[1:6, 1:7] OF INTEGER
# accessing 2D arrays
<arrayIdentifier>[x, y]
# examples
Board[3,4] ← 0 // sets the element in row 3 and column 4 to zero
```

Java:

```
String[] list1 = {"","",""};
int[] list2;
list2 = new int[5];
int[] list3;
list3 = new int[100];
String[] aList;
aList = new String[25];
int[][] board = { {0, 0, 0, 0, 0, 0, 0},
                  {0, 0, 0, 0, 0, 0, 0},
                  {0, 0, 0, 0, 0, 0, 0},
                  {0, 0, 0, 0, 0, 0, 0},
                  {0, 0, 0, 0, 0, 0, 0},
                  {0, 0, 0, 0, 0, 0, 0} }
int[][] board; board = new int[6][7];
//accessing 2D array
board[2][3] = 0;
```

Java:

```
// Writing to a text file
import java.io.FileWriter;
import java.io.PrintWriter;
import java.io.IOException;
FileWriter fileHandle = new
FileWriter("SampleFile.TXT", false);
PrintWriter printLine = new
PrintWriter(fileHandle);
String lineOfText;
printLine.printf("%s"+"%n", lineOfText);
printLine.close();

// Reading from a text file
import java.io.IOException;
import java.io.FileReader;
import java.io.BufferedReader;
FileReader fileHandle = new
FileReader("SampleFile.TXT");
BufferedReader textReader = new
BufferedReader(fileHandle);
String lineOfText = textReader.readLine();
textReader.close();

// Appending to a text file
import java.io.FileWriter;
import java.io.PrintWriter;
import java.io.IOException;
FileWriter fileHandle = new
FileWriter("SampleFile.TXT"), true);
PrintWriter printLine = new
PrintWriter(fileHandle);
String lineOfText;
printLine.printf("%s"+"%n", lineOfText);
printLine.close();

// The end-of-file (EOF) marker
import java.io.IOException;
import java.io.FileReader;
import java.io.BufferedReader;
FileReader fileHandle = new FileReader("Test.
txt");
BufferedReader textReader = new
BufferedReader(fileHandle);
String lineOfText = textReader.readLine();
while (lineOfText != null)
{
   System.out.println(lineOfText);
   lineOfText = textReader.readLine();
}
textReader.close();
```