11. Pseudocode 1

What is pseudocode?

- Pseudocode and flowcharts are both popular ways of representing algorithms.
- Pseudocode is easy to read and write, and allows the programmer to concentrate on the logic of the problem.
- Pseudocode is really structured English. It is English that has been formalised and abbreviated to look like the high-level computer languages.

No standard pseudocode at present

- In general:
  - Statement are written in simple English.
  - Each instruction is written on a separate line.
  - **Keywords** and **indentation** are used to signify particular control structures.
  - Each set of instructions is written from top to bottom, with only one entry and one exit.
  - Groups of statements may be formed into modules, and that module given a name.

How to write pseudocode

- There are **six basic computer operations** and introduces common words and keywords used to represent these operations in pseudocode.

- Each operation can be represented as a straightforward instruction in English, with keywords and indentation to signify a particular control structure.

Operation 1: To receive information

- When a computer is required to receive information or input from a particular source, whether it be a terminal, a disk or any other device, the verbs **Read** and **Get** are used.
  - **Read** is usually used when to receive input from a record on a file
  - **Get** is used when to receive input from the keyboard.

Operation 1: To receive information – cont.

- For example, typical pseudocode instructions to receive information are:

❖ ***Read*** *student_name*

❖ ***Read*** *number_1, number_2*

❖ ***Get*** *system_date*

❖ ***Get*** *tax_code*

- Each example uses a single verb, Read or Get, followed by one or more nouns to indicate what data is to be obtained.

Operation 2: To put out information

- When a computer is required to supply information or output to a device, the verbs **Print, Write, Put, Output or Display** are used in the pseudocode.
  - **Print** is usually used when the output is to be sent to the printer.
  - **Write** is used when the output is to be written to a file.
  - **Put, Output or Display** are used if the output is to be written to the screen.

Operation 2: To put out information – cont.

- Typical pseudocode examples are:

❖ ***Print*** *"Program Completed"*

❖ ***Write*** *customer record to master file*

❖ ***Put*** *name, address and postcode*

❖ ***Output*** *total_tax*

❖ ***Display*** *"End of file"*

Operation 2: To put out information – cont.

- An output **Prompt** instruction is required before an input **Get** instruction. The Prompt verb causes a message to be sent to the screen, which requires the user to respond, usually by providing input. For example:

Prompt *for student_mark*

Get *student_mark*

Operation 3: To perform arithmetic

- Most programs require the computer to perform some sort of mathematical calculation, or to apply a formula, and for these a programmer may use either actual mathematical symbols or the words for those symbols.

Operation 3: To perform arithmetic – cont.

- To be consistent with high-level programming language, the following symbols can be written in pseudocode:

  + for add

  - for subtract

  ***** for multiply

  / for divide

  = indicate assignment of a value as a result of some processing

  ( ) for parentheses

Operation 3: To perform arithmetic – cont.

- For instance, the same pseudocode instructions can be expressed as either of the following:

❖ *add number to total*

❖ *total = total + number*

- The verbs **Compute** and **Calculate** are also available. Some examples are:

divide total_marks by student_count

*sales_tax = cost_price * 0.10*

Compute *C = (F – 32) * 5 / 9*

Operation 3: To perform arithmetic – cont.

- When writing mathematical calculations for the computer, the standard mathematical order of operations applies to pseudocode and to most computer language.
- The first operation carried out will be any calculation contained with parentheses. Next, any multiplication or division, as it occurs from left to right, will be performed. Then, any addition or subtraction, as it occurs from left to right, will be performed.

Operation 4: To assign a value to a variable or memory location

- There are 3 instances in which you may write pseudocode to assign a value to a variable or memory location:
  1. To give data an initial value in pseudocode, the verbs **Initialise** or **Set** are used.
  2. To assign a value as a result of some processing, the symbols **“=”** or **“<-”** are written.
  3. To keep a variable for later use, the verbs **Save** or **Store** are used.

Operation 4: To assign a value to a variable or memory location – cont.

- Some typical pseudocode examples are:

❖ ***Initialise*** *total_price to zero*

❖ ***Set*** *student_count to 0*

❖ *total_price* **=** *cost_price + sales_tax*

❖ *total_price* **<-** *cost_price + sales_tax*

❖ ***Store*** *customer_num in last_customer_num*

- Note the difference of the symbol "=" in operation 3 and 4!

Operation 5: To compare two variables and select one of two alternative actions

- To represent this operation in pseudocode, special keywords are used: **IF, THEN, ELSE, END IF**.
- The comparison of data is established in the **IF** clause, and the choice of alternatives is determined by the **THEN or ELSE** options. Only one of these alternatives will be performed.

Operation 5: To compare two variables and select one of two alternative actions – cont.

- A typical pseudocode example to illustrate this operation is:

```
IF student_attendance_status is part_time THEN
        add 1 to part_time_count
ELSE
        add 1 to full_time_count
END IF
```

Operation 6: To repeat a group of actions

- When there is a sequence of processing steps that need to be repeated, two special keywords, **WHILE…DO…END WHILE**, are used in pseudocode. The condition for the repetition of a group of actions is established in the WHILE clause, and the actions to be repeated are listed beneath DO. For example:

```
WHILE student_total < 50
DO
        Read student record
        Write student name, address to report
        add 1 to student_total
END WHILE
```

Use meaningful variable names

- For example, number1, number2 and number3 are more meaningful names for three numbers than A, B and C.
- If more than one word is used in the name of a variable, then underscores are useful as word separators, for example, sales_tax.
- Most programming languages do not tolerate a space in a variable name.

The three basic control structures

1. **Sequence** - The sequence control structure is the straightforward execution of one processing step after another.
2. **Selection** – The selection control structure is the presentation of a condition and the choice between two actions.
3. **Repetition** – The repetition control structure can be defined as the presentation of a set of instructions to be performed repeatedly, as long as a condition is true.

The selection control structure

- **Simple selection**

IF... THEN
...
ELSE
...
END IF

- **Combined selection**

IF... AND ...THEN
...
END IF

IF... OR ...THEN
...
END IF

- **Nested selection**

IF...THEN
...
ELSE
IF ...THEN
...
ELSE
...
END IF
END IF

The repetition control structure

- Using the **WHILE…DO…END WHILE** structure
- Using the **REPEAT…UNTIL** structure
- Counted **loops**

Some example of pseudocode

Example 1:

- A program is required to read three numbers, add them together and output their total.

Example 1: Possible solution algorithm

```
PROGRAM Add_three_numbers
    Read number1, number2, number3
    total = number1 + number2 + number3
    Output total
END
```

Example 2:

- A program is required to prompt the terminal operator for the maximum and minimum temperature readings on a particular day, accept those readings as integers, and calculate and display to the screen the average temperature.

Example 2: Possible solution algorithm

```
PROGRAM Find_average_temperature
    Prompt for max_temp, min_temp
    Get max_temp, min_temp
    avg_temp = (max_temp + min_temp)/2
    Output avg_temp
END
```

Example 3:

- A program required to read from the screen the length and width of a rectangular house block, and the length and width of the rectangular house that has been built on the block. The algorithm then compute and display the mowing time required to cut the grass around the house, at the rate of two square metres per minute.

Example 3: Possible solution algorithm

```
PROGRAM Calculate_mowing_time
        Prompt for block_length, block_width
        Get block_length, block_width
        block_area = block_length * block_width

        Prompt for house_length, house_width
        Get house_length, house_width
        house_area = house_length * house_width

        mowing_area = block_area – house_area
        mowing_time = mowing_area / 2
        Output mowing_time
END
```

Example 4: Use nested selection/Case

- A program is required to read a customer's name, a purchase amount and a tax code. The tax code has been validated and will be one of the following:
  - 0 tax exempt (0%)
  - 1 sales tax (17%)
  - 2 special sales tax (20%)

  The program must then compute the sales tax and the total amount due, and print the customer's name, purchase amount, sales tax and total amount due.

Example 4: Possible solution algorithm - 1

```
PROGRAM Process_customer_record
    Read cust_name, purch_amt, tax_code
    IF tax_code = 0 THEN
       sales_tax = 0
    ELSE
        IF tax_code = 1 THEN
           sales_tax = purch_amt * 0.17
        ELSE
           sales_tax = purch_amt * 0.2
        END IF
     END IF
     total_amt = purch_amt + sales_tax
     Print cust_name, purch_amt, sales_tax, total_amt
END
```

Example 4: Possible solution algorithm - 2

```
PROGRAM Process_customer_record
    Read cust_name, purch_amt, tax_code
    CASE OF tax_code
        0: sales_tax = 0
        1: sales_tax = purch_amt * 0.17
        2: sales_tax = purch_amt * 0.2
     END CASE
    total_amt = purch_amt + sales_tax
    Print cust_name, purch_amt, sales_tax, total_amt
END
```

References

- 2007, Lesley Anne Robertson, Simple Program Design A Step-by-Step Approach, Fifth edition.