

Pseudocode

Pseudocode is informal language that helps programmers develop algorithms (or recipes). An algorithm is a procedure for solving a problem.

An algorithm describes actions to be executed and the order in which those actions are to be executed. In other words, an algorithm is merely the sequence of steps taken to solve a problem; like a recipe. An algorithm is not computer code. Algorithms are just the instructions which provide a clear path for you to write the computer code.

For example, below is an algorithm, written in basic pseudocode, for adding two numbers together:

Step 1: Start

Step 2: Declare variables num1, num2, and sum.

Step 3: Read values num1 and num2.

Step 4: Add num1 and num2 and assign the result to sum.

Step 5: Display sum

Step 6: Stop

It is important to note that the algorithm is the step-by-step instructions for accomplishing a task, while pseudocode refers to how it was written and described.

Algorithm	Pseudocode
An unambiguous specification of how to solve a problem.	An informal high-level description of the operating principle of a computer program or other algorithm.
Helps to simplify and understand the problem.	A method of developing an algorithm.

Although there are no hard and fast rules for pseudocode, there are some suggestions to help make pseudocode more understandable and easy to read.

For instance, consider indenting all statements showing a “dependency”, like statements that use: While, do, for, if.

Example 1.

Pseudocode:

If students grade is higher than or equal to 60

 Then Print, “Passed”

else

 Print, “Failed”

The above pseudocode would be used to develop the following C++ code.

C++ Source Code:

```
if(grade > 60 || grade == 60)
{
    cout << "Passed" << endl;
}
else
{
    cout << "Failed" << endl;
}
```

Example 2.

Pseudocode:

Set total to zero

initialize and type variables

While grade counter is less than or equal to ten

 Input the next grade score

 Add the grade score into the total

Set the class average to the total divided by ten

Print the class average.

C++ Source Code:

```
double total = 0;
double grade = 1;
double score = 0;
```

```

while(grade <= 10)
{
    cout << "Enter a score: ";
    cin >> score;
    cout << endl;
    total = total + score;
    grade++;
}
double average;
average = total / 10;
cout << average;

```

Example 3.

Pseudocode:

Initialize total to zero

Initialize counter to zero

Input the first grade

While the user has not as yet entered the stop code

 Add this grade into the running total

 Add one to the grade counter

 Input the next grade (possibly the sentinel)

endwhile

If the counter is not equal to zero

 Set the average to the total divided by the counter

 Print the average

Else

 Print, "No grades were entered"

Endif

C++ Source Code:

```

double total = 0;
double counter = 0;

double grade = 0;
cout << "Enter a grade. Enter 999 to quit: ";
cin >> grade;
cout << endl;

while(grade != 999)

```

```

{
    total = total + grade;
    counter++;

    cout << "Enter another grade. Enter 999 to quit: ";
    cin >> grade;
}

if(counter != 0)
{
    double average;
    average = total / counter;
    cout << average << endl;
}
else
{
    cout << "No grades were entered." << endl;
}

```

Example 4.

pseudocode:

Read the length of the rectangle

Read the width of the rectangle

Compute the area of the rectangle as length times width.

C++ Source Code:

```

double length = 0;
double width = 0;

cout << "What is the rectangle length?: " << endl;
cin >> length;

cout << "What is the rectangle width?: " << endl;
cin >> width;

cout << "The rectangle area is: " << length * width;

```

Common Action Keywords for Pseudocode

Several keywords are often used to indicate common input, output, and processing operations.

Input: **READ, OBTAIN, GET**

Output:	PRINT, DISPLAY, SHOW
Compute:	COMPUTE, CALCULATE, DETERMINE
Initialize:	SET, INITIALIZE
Add one:	INCREMENT, BUMP

Some Keywords:

For looping and selection, The keywords that you might consider writing include:

Do While...

Do Until...

Case...

If... then...

Call ... with (parameters)

Call

Return

Return

When

Try to indicate the end of loops and iteration by using scope terminators.

For instance use **if... (statements) ... endif**.

As verbs, consider using the words:

Generate, Compute, Process, set, reset, increment, compute, calculate, add, sum, multiply, subtract, divide, print, display, input, output, edit, test, etc.

Be sure to indent if the indentation fosters understanding.

Being clear is the purpose of pseudocode, and a very desirable goal to strive for.