Lab 7: Design computer Algorithms

Material

An algorithm (pronounced AL-go-rith-um) is a procedure or formula for solving a problem. The word derives from the name of the mathematician, Mohammed ibn-Musa al-Khwarizmi, who was part of the royal court in Baghdad and who lived from about 780 to 850. Al-Khwarizmi's work is the likely source for the word algebra as well.

To make a computer do anything, you have to write a computer program. To write a computer program, you have to tell the computer, step by step, exactly what you want it to do. The computer then "executes" the program, following each step mechanically, to accomplish the end goal.

When you are telling the computer what to do, you also get to choose how it's going to do it. That's where computer algorithms come in. The algorithm is the basic technique used to get the job done. Let's follow an example to help get an understanding of the algorithm concept.

Let's say that you have a friend arriving at the airport, and your friend needs to get from the airport to your house. Here are four different algorithms that you might give your friend for getting to your home:

The taxi algorithm:

1. Go to the taxi stand.
2. Get in a taxi.
3. Give the driver my address.

The call-me algorithm:

1. When your plane arrives, call my cell phone.
2. Meet me outside baggage claim.

The rent-a-car algorithm:

1. Take the shuttle to the rental car place.
2. Rent a car.
3. Follow the directions to get to my house.

The bus algorithm:

1. Outside baggage claim, catch bus number 70.
2. Transfer to bus 14 on Rukab Street.
3. Get off on Jerusalem street.

4. Walk two blocks north to my house

All four of these algorithms accomplish exactly the same goal, but each algorithm does it in completely different way. Each algorithm also has a different cost and a different travel time. Taking a taxi, for example, is probably the fastest way, but also the most expensive. Taking the bus is definitely less expensive, but a whole lot slower. You choose the algorithm based on the circumstances.

Pseudocode

Pseudocode is a kind of structured English for describing algorithms. It allows the designer to focus on the logic of the algorithm without being distracted by details of language syntax. At the same time, the pseudocode needs to be complete. It describes the entire logic of the algorithm so that implementation becomes a rote mechanical task of translating line by line into source code.

Examples#1:
Calculate the class average for 10 students:

Pseudocode:

```
Set total to zero
Set counter to one
While counter is less than or equal to ten
    Input the next grade
    Add the grade into the total
ENDWhile
Set the average to the total divided by ten
Print the class average
```

Examples#2:

Sorting

Sorting means ordering the elements is ascending or descending order.

Bubble sort is a simple sorting algorithm. It works by repeatedly stepping through the list to be sorted, comparing two items at a time and swapping them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. The algorithm gets its name from the way smaller elements "bubble" to the top of the list. Because it only uses comparisons to operate on elements, it is a comparison sort

Let us take the array of numbers "5 1 4 2 8", and sort the array from lowest number to greatest number using bubble sort algorithm. In each step, elements written in bold are being compared.

First Pass:

(5 1 4 2 8) ⟶(1 5 4 2 8) Here, algorithm compares the first two elements, and swaps them.
(1 5 4 2 8) ⟶(1 4 5 2 8)
(1 4 5 2 8) ⟶(1 4 2 5 8)
(1 4 2 5 8) ⟶(1 4 2 5 8) Now, since these elements are already in order, algorithm does not swap them.
Second Pass:

(1 4 2 5 8) ⟶(1 4 2 5 8)
(1 4 2 5 8) ⟶(1 2 4 5 8)
(1 2 4 5 8) ⟶(1 2 4 5 8)
(1 2 4 5 8) ⟶(1 2 4 5 8)
Now, the array is already sorted, but our algorithm does not know if it is completed. Algorithm needs one whole pass without any swap to know it is sorted.

Third Pass:

(1 2 4 5 8) ⟶(1 2 4 5 8)
(1 2 4 5 8) ⟶(1 2 4 5 8)
(1 2 4 5 8) ⟶(1 2 4 5 8)
(1 2 4 5 8) ⟶(1 2 4 5 8)
Finally, the array is sorted, and the algorithm can terminate...

A simple way to express bubble sort in pseudocode is as follows:

```
procedure bubbleSort( A : list of sortable items ) defined as:
  do
```

```
    swapped := false
    for each i in 0 to length( A ) - 2 do:
      if A[ i ] > A[ i + 1 ] then
        swap( A[ i ], A[ i + 1 ] )
        swapped := true
      end if
    end for
  while swapped
end procedure
```

EXERCISES

1- Write an algorithm to calculate the average of a set of students (we don't know their count).

2- Write an algorithm to print the number of passes and the number of failures in a class of n students, also let the program print the failure percentage.

3- Design an algorithm that will prompt for and receive prices of several items. After the last price is entered, the sentinel amount of –1 is entered. The algorithm should calculate the number of items purchased, total cost of the purchase before tax and with the tax of 7.5%, and display the results on the screen.

4- Write an Algorithm to do the function of a simple calculator which should be able to do +,-,*,% operations.

5- Write an algorithm to print the sum of the given series, take first 8 terms

$A=1! +2! +3! +4! +5! +\ldots$

6- Find the maximum and the minimum elements for a set of n integers.