Computer Fundamentals

Examples of Algorithm, Pseudo code and Flowchart

Professor Dr. M. Ismail Jabiullah
Professor
Department of CSE
Daffodil International University
Bangladesh

Example of Algorithm, Pseudo Code and Flowchart

Topics

- Problem Analysis
- Algorithm
- Pseudo code
- Examples
- Flowcharts
- Examples of Flowcharts

Algorithm and Pseudo Code

Example - 1

Problem 1: Given a list of positive numbers, return the largest number on the list.

Inputs: A list L of positive numbers. This list must contain at least one number. (Asking for the largest number in a list of no numbers is not a meaningful question.)

Outputs: A number n, which will be the largest number of the list.

Algorithm:

1. Set max to 0.
2. For each number x in the list L, compare it to max.
3. If x is larger, set max to x.
4. max is now set to the largest number in the list.

Program Segment

```
def find_max (L):
    max = 0
    for x in L:
        if x > max:
            max = x
    return max
```

Algorithm and Pseudo Code

Example - 2

Problem 2: A Recursive Version of find_max()

Inputs: A list L of positive numbers. This list must contain at least one number. (Asking for the largest number in a list of no numbers is not a meaningful question.)

Outputs: A number n, which will be the largest number of the list.

There can be many different algorithms for solving the same problem. Here's an alternative algorithm for **find_max()**:

1. If L is of length 1, return the first item of L.
2. Set v1 to the first item of L.
3. Set v2 to the output of performing **find_max()** on the rest of L.
4. If v1 is larger than v2, return v1. Otherwise, return v2.

Program Segment

```
def find_max (L):
if len(L) == 1: return L[0]
v1 = L[0]
v2 = find_max(L[1:])
if v1 > v2: return v1
else:   return v2
```

Algorithm and Pseudo Code

Example - 3

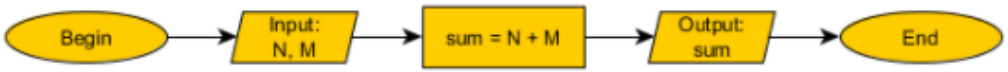
Problem 2: Find the sum two numbers N and M.

The procedure is:

1. Enter the two numbers in the variables N and M.
2. Sum them and save the result in the variable sum.
3. Output the result

Program Segment

```
def find_max (L):
if len(L) == 1: return L[0]    v1 = L[0]
v2 = find_max(L[1:])
if v1 > v2: return v1
else:   return v2
```



Algorithm

Algorithm 1: Add two numbers entered by the user

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.
sum←num1+num2
Step 5: Display sum
Step 6: Stop

Algorithm

Algorithm 2: Find the largest number among three numbers

Step 1: Start
Step 2: Declare variables a, b and c.
Step 3: Read variables a, b and c.
Step 4: If a > b If a > c
    Display a is the largest number.
  Else
    Display c is the largest number.
  Else
    If b > c
      Display b is the largest number.
    Else
      Display c is the greatest number.
Step 5: Stop

Algorithm

Algorithm 3: Find Root of the quadratic equation $ax^2 + bx + c = 0$

Step 1: Start
Step 2: Declare variables a, b, c, D, x1, x2, rp and ip;
Step 3: Calculate discriminant D ← b2-4ac
Step 4: If D ≥ 0
    r1 ← (-b+√D)/2a
    r2 ← (-b-√D)/2a
    Display r1 and r2 as roots.
  Else
    Calculate real part and imaginary part
    rp ← -b/2a
    ip ← √(-D)/2a
    Display rp+j(ip) and rp-j(ip) as roots
Step 5: Stop

Algorithm

Algorithm 4: Find the factorial of a number

Step 1: Start
Step 2: Declare variables n, factorial and i.
Step 3: Initialize variables factorial ← 1 i ← 1
Step 4: Read value of n
Step 5: Repeat the steps until i = n
    5.1: factorial ← factorial*i
    5.2: i ← i+1
Step 6: Display factorial
Step 7: Stop

Algorithm

Algorithm 5: Check whether a number is prime or not

Step 1: Start
Step 2: Declare variables n, i, flag.
Step 3: Initialize variables flag ← 1 i ← 2
Step 4: Read n from the user.
Step 5: Repeat the steps until i=(n/2)
    5.1 If remainder of n÷i equals 0 flag ← 0 Go to step 6
    5.2 i ← i+1
Step 6: If flag = 0 Display n is not prime else Display n is prime
Step 7: Stop

Algorithm

Algorithm 6: Find the Fibonacci series till the term less than 1000

Step 1: Start
Step 2: Declare variables first_term,second_term and temp.
Step 3: Initialize variables first_term ← 0 second_term ← 1
Step 4: Display first_term and second_term
Step 5: Repeat the steps until second_term ≤ 1000
    5.1: temp ← second_term
    5.2: second_term ← second_term + first_term
    5.3: first_term ← temp
    5.4: Display second_term
Step 6: Stop

Algorithm

Algorithm 5: create an algorithm to check whether a number is positive or negative.

Step 1: Start
Step 2: Print “Give any number”.
Step 3: Read num
Step 4: If (num==0) Print “You Entered 0”
Step 5: If (num>0) Print “You Entered a positive number”
Step 6: If (num<0) Print “You Entered a Negative number”
Step 7: Stop

1. Print “Give any number”
2. Read num
3. if (num==0) print “You entered 0”
4. if (num>0) print “You entered a positive number”
5. if (num<0) print “You entered a negative number”

Pseudo code

Pseudo code 1: create an algorithm to check whether a number is positive or negative.

1. If student's grade is greater than or equal to 60
2. Print "passed"
3. else
4. Print "failed"

Pseudo code

Pseudo code 2: create an algorithm to check whether a number is positive or negative.

1. Set total to zero
2. Set grade counter to one
3. While grade counter is less than or equal to ten
4. Input the next grade
5. Add the grade into the total
6. Set the class average to the total divided by ten
7. Print the class average.

Pseudo code

Pseudo code 3: create an algorithm to check whether a number is positive or negative.

1. Initialize total to zero
2. Initialize counter to zero
3. Input the first grade
4. while the user has not as yet entered the sentinel
5. add this grade into the running total
6. add one to the grade counter
7. input the next grade (possibly the sentinel)
8. if the counter is not equal to zero
9. set the average to the total divided by the counter
10. print the average
11. else
12. print 'no grades were entered'

Pseudo code

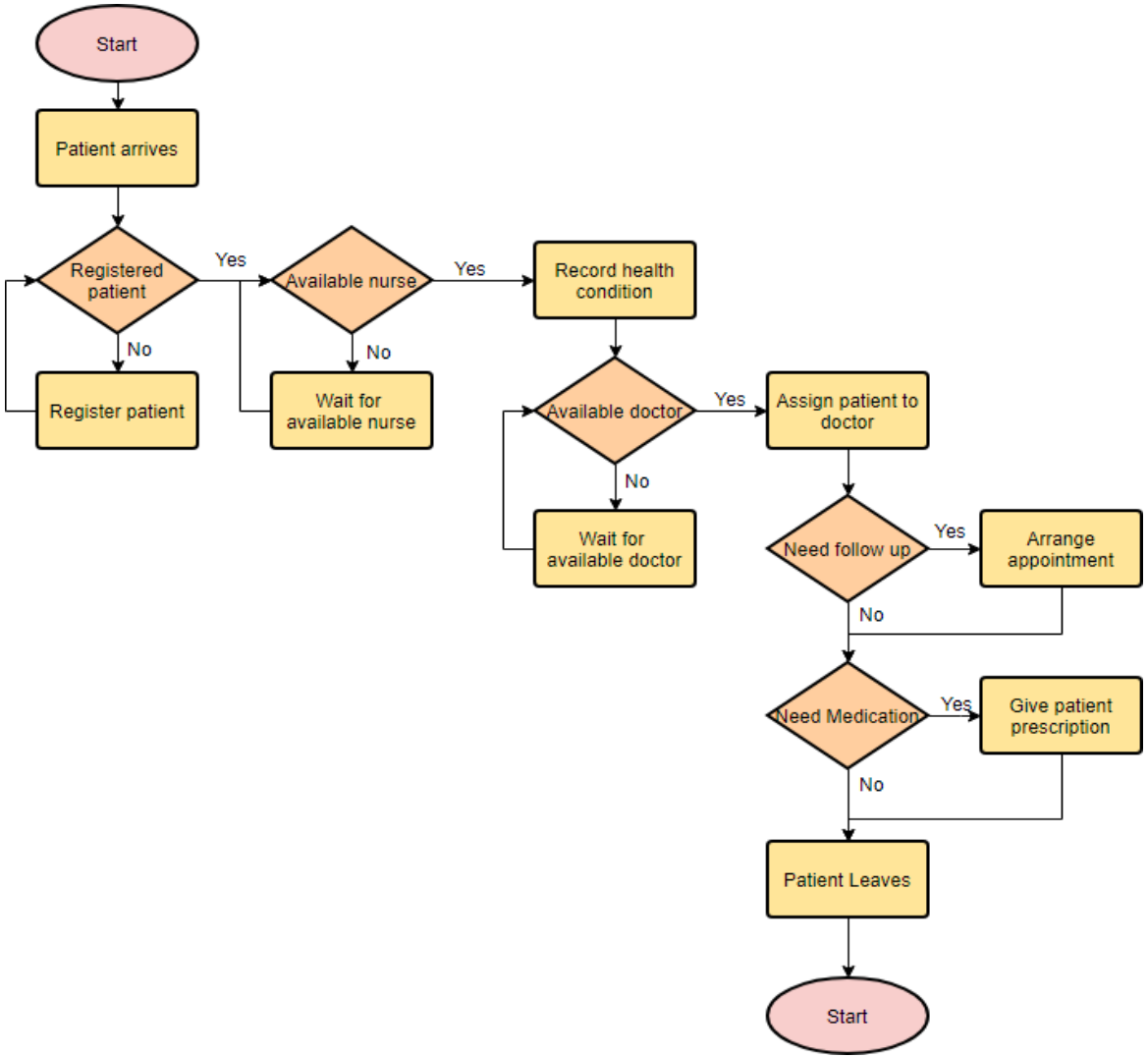
Pseudo code 4: create an algorithm to check whether a number is positive or negative.

1. initialize passes to zero
2. initialize failures to zero
3. initialize student to one
4. while student counter is less than or equal to ten
5. input the next exam result
6. if the student passed add one to passes
   else add one to failures
   add one to student counter
7. print the number of passes
8. print the number of failures
9. if eight or more students passed print "raise tuition"

Flowcharts

Medical Service

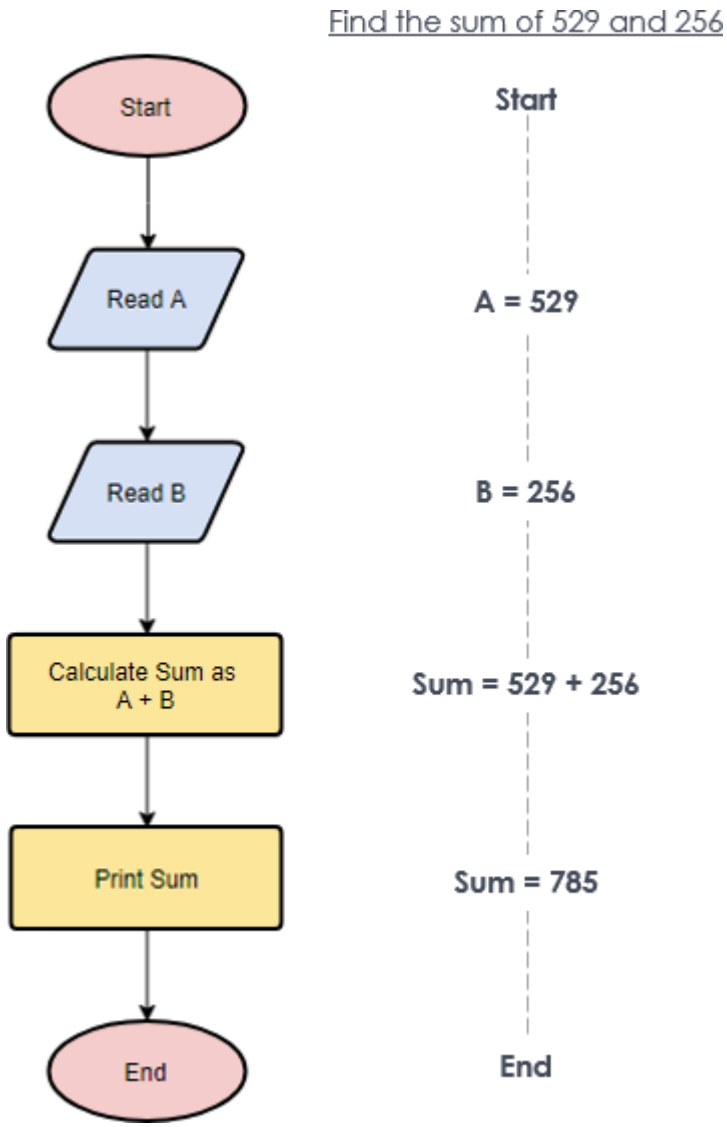
This is a hospital flowchart example that shows how clinical cases shall be processed. This flowchart uses decision shapes intensively in representing alternative flows.



Flowcharts

Sum of Two Numbers

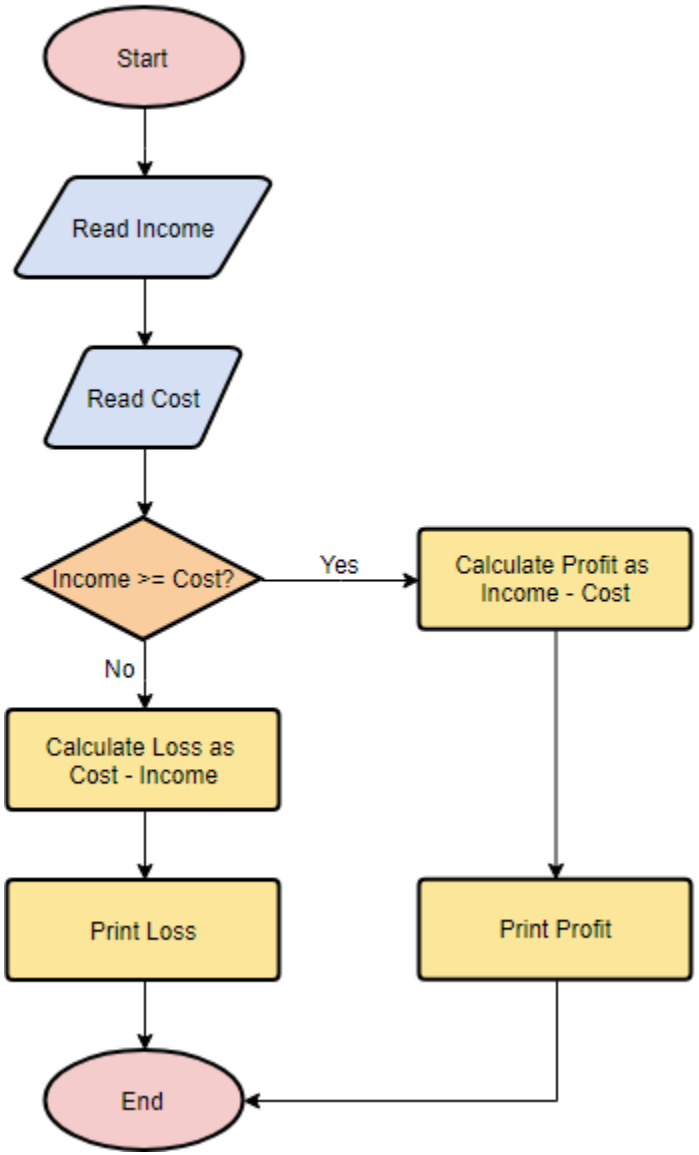
Read two numbers 529 and 256 from the keyboard and find and display the sum of the two numbers.

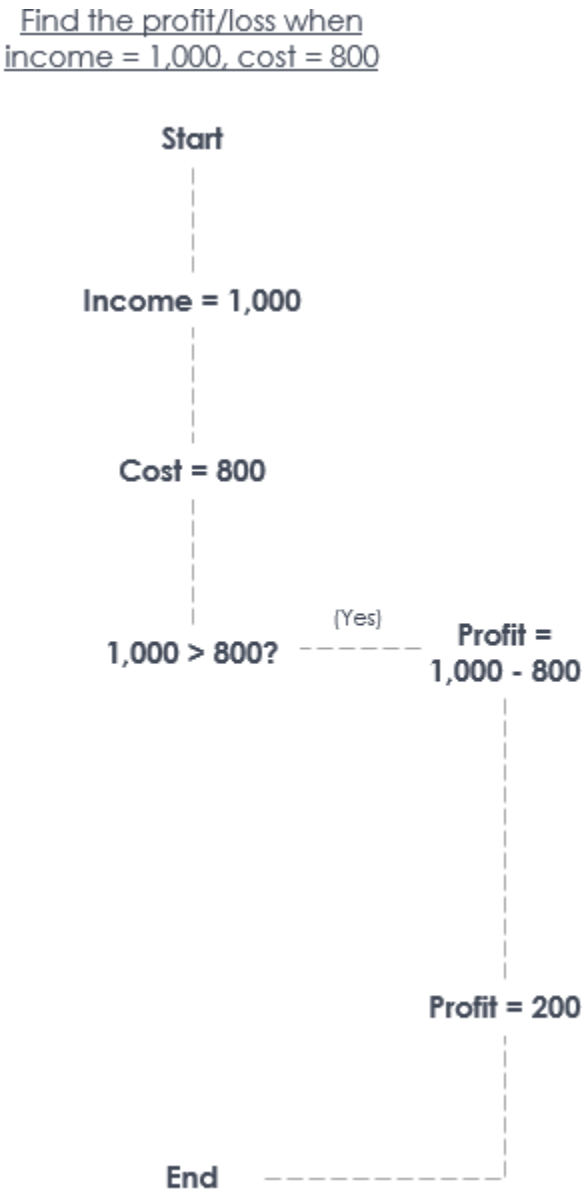


Flowcharts

Profit and Loss Calculations

Read Income and Cost, and Calculate the Profit and Loss.
Finally Show them.





Thanks