Programming Foundation with Pseudocode - Lab Book

Document Revision History

Date	Revision No.	Author	Summary of Changes
Jan 2015	1.0	Rathnajothi Perumalsamy	Initial Draft
May 2015	1.1	Shraddha P Jadhav	Added assignments related to algorithm design and techniques.
May 2016	1.2	Anjulata Tembhare	Removed some assignments as per new ToC

Table of Contents

Getting Started

Overview

This lab book is a guided tour for learning Programming Foundation with Pseudocode. It comprises 'To Do' assignments. Follow the steps provided and work out the 'To Do' assignments given.

Setup Checklist for Programming Foundation with Pseudocode for All LOTs

Here is what is expected on your machine in order for the lab to work.

Minimum System Requirements

- Intel Pentium 90 or higher (P166 recommended)
- Microsoft Windows 95, 98, or NT 4.0, 2k, XP, 7.
- Memory: 32MB of RAM (64MB or more recommended)

Please ensure that the following is done:

- An editor like Notepad/Notepad ++ is installed.

Instructions

- For all instructions on Programming Foundation lab assignment refer Appendix A. All lab assignments should refer these set of instructions.
- Create a directory by your name in drive <drive>. In this directory, create a subdirectory programming_foundation_assgn. For each lab exercise create a directory as lab <lab number>.

Lab 1. Basic program development with pseudocode

Goals	Test the basic programming concept.
Time	4 Hours

1 Write a pseudocode to accept 2 numbers and find the difference between those numbers.

Solution:
Step 1: Open notepad, write the below pseudocode and save the file as "Difference.txt"

```
BEGIN
        DECLARE num, num1 AS INTEGER
        PROMPT "Enter 2 numbers" AND STORE IN num, num1
        PRINT "The result is", num1-num2;
END
```

2 Write a pseudocode to accept 10 numbers and find maximum number among 10 numbers.

Solution:

```
BEGIN
    DECLARE numbers[10] AS INTEGER ARRAY
    DECLARE max AS INTEGER
    INITIALIZE max TO 0
        PROMPT "Enter 10 numbers"
    FOR index=1 TO 10
        ACCEPT numbers[index]
    END FOR
    max=numbers[0]
    FOR index=1 TO 10
        IF numbers[index] > max THEN
            max=numbers[index]
        END IF
    END FOR
    PRINT max
END
```

Assignments: <<TODO>>

1.1 Rima is working in State Electricity Board Project. She got the following requirement. The following information has to be accepted from the user to calculate the net electricity bill amount.

- User ID
- User Name
- Last month meter reading
- Current month meter reading

Unit consumed = (Last month meter reading) – (Current month meter reading)
Net Amount = Unit consumed * 1.15 +Fixed Charge
Assume Fixed Charge is always Rs.100.
Write pseudo code to print the electricity bill. The bill should be in the following format
User ID:
User Name:

Unit Consumed:
Net amount:

1.2 Organization employees are recognized with different tags based on their experience.

Years of Experience	Tag Color
0 - <3	Blue
3 - <5	Grey
5 - <10	Yellow
>10	Red

Write pseudo code to accept the experience and display their tag Color.

1.3 Write pseudo code to print the following mathematical series.
0 1 1 2 3 5 8 13 21 …. N. Where N is accepted from the user.

1.4 Write a pseudo code to accept a number and check whether a given number is an Armstrong number. For example, 371 is an Armstrong number since (3*3*3) + (7*7*7) + (1*1*1) = 371.

1.5 Write a pseudo code to convert a binary number to decimal. For an Example, if the given input is 00000100(binary number), then the output should be 4(Decimal number).

1.6 Write a pseudo code to accept 10 numbers in an array and do the following using a loop.

1.6.1 Display the smallest number.
1.6.2 Display all ODD and EVEN numbers separately

1.7 Modify the below Pseudocode to implement good programming practices. The below Pseudocode is used to calculate total price of a product including tax.

```
BEGIN
    Print "Enter price of your product"
    Accept p
    tc=p*.56
    Print "Total price of product is":  tc
END
```

Hint: .56 is the Tax rate.

Lab 2. Implementation of good programming practices

Goals	Test the concept of applying the characteristics of good programming practices like modularity approach.
Time	3 Hours

1 Write a pseudocode to accept 2 numbers and find the difference between two given numbers

Solution:

```
BEGIN
                    DECLARE num, num1 AS INTEGER
                    PROMPT "Enter 2 numbers" AND STORE IN num, num1
                    difference (num,num1);
 END
 SUB difference (num1, num2)
                    PRINT "The result is  " , num1-num2;
 END SUB
```

Assignments: <<TODO>>

2.1 Modify all the pseudo code written in lab 1 to ensure that the code adheres to good programming practices such as readability and maintainability.

2.2 Write a module in pseudo code to accept a number and return the sum of its digits. Invoke the user defined module in main program. For an example: if a user enters input as 12, then the output will be 3.

2.3 Write a module in pseudo code to accept an array with 10 numbers and return number of unique values in an array. For an example, if the given input array contains values such as {10,22,13,22,44,6,24,44,77,8}, then the output should be 6.

2.4 Refactor the below given code. The below pseudocode is used for calculating total leaves applicable per year for an employee. Number of Leaves added to an employee account differs based on the type of employment. For an Example, 2 days leave will be added per month for a permanent employee and 1 day leave will be added per month for a contract based employee

```
RECORD Employee
    DECLARE EmpId as INTEGER
    DECLARE employmentType AS STRING
END RECORD

BEGIN
                    DECLARE emp AS Employee
            //leaves variable is used to store number of leaves per month
            DECLARE leaves, TotalLeaves AS INTEGER
            FOR index= 1 to 5
                    PROMPT "Enter the EmployeeId" AND STORE IN
        emp.EmpId
                    PROMPT "Enter the employmentType" AND STORE IN
    emp.employmentType
            END FOR
```



```
                    FOR index= 1 to 5
                        IF(employmentType=='PERMANENT') THEN
                            leaves=2;
                            TotalLeaves=leaves*12;
                        ELSE IF(employmentType=='CONTRACT') THEN
                            leaves=1;
                            TotalLeaves=leaves*12;
                        END IF
                            PRINT "Employee Id :" , emp.EmpId
                          PRINT "Total Available Leaves are :", TotalLeaves
                    END FOR
                    Index2=5;
END
```


Lab 3. Algorithm Analysis and Design

Goals	Algorithm analysis and Test the concept of searching and sorting algorithms.
Time	3 Hours

Assignments: <<TODO>>

3.1 **ALGORITHM Check(A[0..n-1])**

```
//Input: An array A[0..n-1] of n real numbers
x← A[0]; y←A[0]
for i←1 to n-1 do
    if A[i]<x
        x← A[i]
    if A[i]>y
        y← A[i]
return y-x
```

For the algorithm given above,

- What does this algorithm computes?
- What is its basic operation?
- Check the no of times the basic operation is executed depends only on the size of an input or on any other parameter.
- Set up a sum/recurrence, expressing the number of times the basic operation is executed.
- What is the efficiency class of the given algorithm?

3.2 Determine the output of below given algorithm by tracing it for the taking any sample input.

ALGORITHM surprise(A[0..n-1])

```
//Input: An array A[0..n-1] of real numbers
If n=1
    return A[0]
else
    t←surprise(A[0..n-2])
    if t<=A[n-1]
        return t
    else
        return A[n-1]
```

3.3 Find the time efficiency class of the following algorithm,

```
ALGORITHM Find(Matrix[0..n-1,0..n])
//Input: An n by n+1 matrix Matrix[0..n-1,0..n] of real numbers
for index← 0 to n-2 do
    for nextindex← index+1 to n-1 do
        for tempindex← index to n do

            Matrix[nextindex,tempindex]←Matrix[nextindex,tempindex]-
Matrix[index,tempindex] * Matrix[nextindex,index] / Matrix[index,index]
```

3.4 Consider the following Algorithm,


```
ALGORITHM Sum(n)
//Input: A nonnegative integer n
sum←0
for index←1 to n do
     sum←sum+i
return sum
```

- What does the above algorithm compute?
- What is the efficiency class of the given algorithm?

3.5 Write a module in pseudo code with 2 parameters: first parameter is a word, the second is a character. The function must return the number of times the character is found in the word.

3.6 Write a Pseudocode to accept 5 numbers in an array and sort the array if array is not sorted and search for a number in the array using binary search. Implement sorting and searching logic in module.

<<Stretched Assignments>>

3.7 Write a module in pseudo code to accept 2 strings and return success message if second string is a substring of first string. For Example: If a user enters first string as “Testing” and second string as “Test”, then “Substring found in source string” message should be displayed.

Lab 4. Exception Handling

Goals	Test the concept of handling exceptional scenarios.
Time	1.5 Hours

4.1 Write a pseudo code for calculating price after applying discount in which exceptions handlers are used.

Solution:

```
/**************************************************************************
 * File                         : DefensiveProgramming.txt
 * Author Name                          : Capgemini
 * Desc                         : Program to apply discount on productprice
 * Version                      : 1.0
 * Last Modified Date           : 8-May-2016
 * Change Description           : Description about the changes implemented
 **************************************************************************/

BEGIN
  DECLARE productId AS INTEGER
  DECLARE discount AS REAL
  PROMPT "Enter productId" AND STORE IN productId
  IF(isValid(productId)) THEN
      PROMPT "Enter discount" AND STORE IN discount
      IF(isValid(discount)) THEN
              applyDiscount(productId,discount);
      ELSE
              PRINT"Discount value should contain numbers"
      END IF
  ELSE
      PRINT"Product Id should contain numbers"
  END IF
END

/**************************************************************************
* Module Name                  : applyDiscount()
* Input Parameters    : productId, discount
* Return Type          : INTEGER
* Author                : Capgemini
* Creation Date        : 8-May-2016
* Description          : Applying discount on the product price
 **************************************************************************/

SUB applyDiscount(productId,discount)
  DECLARE result AS INTEGER
  result=getProductPrice(productId)*discount
  PRINT "Product Price" + result;
EXCEPTION
  WHEN NoSuchElement THEN
     PRINT errormessage //Errormessage returned from exception
END SUB

/**************************************************************************
* Module Name                  : getProductPrice()
* Input Parameters    : productId
```



```
* Return Type        : INTEGER
* Author             : Capgemini
* Creation Date      : 8-May-2016
* Description        : Based on productId, fetching product price if productId exists,
else exception will be raised
 **********************************************************************/

SUB getProductPrice(productId)
 DECLARE errorcode AS INTEGER AND STORE 0
 IF(elementfound(productId)) THEN
       RETURN productPrice
 ELSE
    RAISE "Product doesn't exist with this id"+ productId
 END IF
END SUB

/**********************************************************************
* Module Name                : isValid()
* Input Parameters   : data
* Return Type        : BOOLEAN
* Author             : Capgemini
* Creation Date      : 8-May-2016
* Description        : To validate data for accepting only digits
**********************************************************************/

SUB isValid(data)
 IF(isDigits(data)) THEN
    RETURN true
 ELSE
    RETURN false
 END IF
END SUB
```

Assignments: <<TODO>>

4.2 Take care of necessary validations in 2.4 assignment in order to avoid exceptional scenarios

4.3 Modify the below Pseudocode to implement good programming practices and take care of necessary validations to avoid exceptional scenarios. The below Pseudocode is used to calculate total price of a product including tax.

```
BEGIN
    Print "Enter price of your product"
    Accept p
    Print "Enter quantity of your product"
    Accept q
    tc=p*q*.56
    Print "Total price of product is":  tc
END
```

Hint: .56 is the Tax rate.

Lab 5. Software Review and Testing

Goals	Test the concept of reviews and testing
Time	1.5 Hours

Assignments: <<TODO>>

5.1 Do self and peer review of the pseudo code for previous lab assignments 1.2, 2.4 and 3.4 using the review check list and record the review comments in the checklist.

5.2 Write test cases for Lab assignment 1.2, 2.4 and 3.7 using the test case template

Appendices

Appendix A: Programming Standards

Instructions for lab assignments:

Compile the list of learning's from the previous day.

a. Maintain the list of steps required to complete each type of task. For example the following steps are required to develop a new piece of code:

   i. Study the Specifications or Problem Definition (Analysis).
   ii. Develop the Program Design or Pseudo Code (Design).
   iii. Develop the Black Box Test cases in the specified format (Testing).
   iv. Develop the Documented Code (Coding).
   v. Develop the White Box Test cases in the specified format (Testing).
   vi. Perform the Code Review - Self and/or Peer (Reviews).
   vii. Note the Number and Category of defects found. Categories are: Standards (Program output not affected), Logic (Program output would differ)
   viii. Compile and remove compilation errors (Testing).
   ix. Test and log all defects found (Testing).
   x. Debug and fix all defects found (Debugging).
   xi. Repeat steps xi and xii, as required. However, note the number of iterations.
   xii. Note the Time / Efforts spent on each step above. The item in brackets indicates the category against which Time / Efforts should be logged.
   xiii. Compare the Actual Time against the estimated time. Note down reasons for significant variations.

b. Maintain a check-list of "Things-To-Do", or "Things-To-Check-for" (includes "Things-NOT-To-Do"). Use this check-list during Code Review and Test Case Review. Revise the check-list every day based on the learnings of the previous day.

c. For all assignments, the test cases should be created in the following format by using an EXCEL sheet:

   i. Requirement Id #, Test Case Id #, Test condition, Test cases, Test data, Expected result, Remarks

d. Defects should be logged in the following format:

   i. Bug #, Failed test case #, Bug description, Bug Status, Fixed in Iteration #, Time/Efforts spent on debugging, Type of Error

e. Time / Efforts should be logged against the following categories:
   i. Analysis, Design, Coding, Reviews, Testing, and Debugging
   Check the % of time spent in different categories.

Sequence of assignments may be adjusted by the faculty based on performance of participants.

