RATIONALE

Why This Module

The Programming Logic and Techniques (PLT) module is positioned as the entry point into this curriculum. Since this curriculum is designed to cater to aspiring software professionals, a strong foundation in programming approaches and application of logic was found essential and was recommended by the industry.

Linkage with OOPs

In the OOPs module the students will be required to write programs involving the use of programming constructs. The PLT module, apart from teaching problem solving technique like flowcharting, will also prepare the students for the OOPs module by developing the students ability to study problems, understand problems, and represent the logic required to solve these problems.
The objective of this module is to familiarize the students with the basic constructs and techniques used in solving problems.

Since the entry profile does not expect the student to have any exposure to programming the module starts from the very basics of flowcharting and takes the student gradually through higher levels of problem solving through flowcharts.
ENTRY PROFILE

A student who registers for the PLT module should be:

- Familiar with the various parts of the computer and should know the basic difference between hardware and software.

- Able to interact in English in a classroom environment because the classes will be conducted in English. The courseware will also be in English. The sessions will be interactive in nature.

- Comfortable working in the Windows environment. Knowledge of MS Office would be an added advantage.
At the end of this module, the students will be able to:

- Create flowcharts to solve problems.

Knowledge acquired:

- Symbols used in flowcharting
- Sequencing the tasks to be carried out
- Decision making using flowcharts
- Iteration using flowcharts
- Identification of
  - Variables and constants
  - Data types
- Organizing and carrying out
  - Conditional execution
  - Iteration
- Implementing procedures
- Concept of a dry run
NOTE FOR FACULTY

As faculty you might find the contents of this module are light and may feel the need for greater depth of coverage. Please remember that this is the first exposure to programming for most students and we must concentrate on ensuring the following:

- They are very confident of analyzing problems and developing solutions using the constructs learned
- They find it a very enjoyable and enriching experience
- They have had a taste of ‘programming’, and look forward to more of it in the OOPs module

Remember, once they start enjoying drawing flowcharts and learning the logic of programs they will be sufficiently motivated to put in more effort to learn more in the further modules. Therefore, if you get them hooked right here, you have won half your battle.

Make sure that you do not go overboard in giving inputs. Let them apply logic from their everyday experience.

Prevent yourself from lapsing back into the topic-centered approach. Focus on tasks, and the knowledge required for accomplishing those tasks.
Lesson One

Experiences

During the Analysis phase of courseware development, we conducted several interviews with project managers in the software industry. They told us that they found programming logic and skills lacking in many of the new recruits. They suggested that this module needs to be given a lot of emphasis.

People wanted to know why are we teaching flowcharts. Many people are under the impression that flowcharts are not used in the software industry. But the fact revealed during our analysis confirmed that flowcharts are used extensively in software development.

Examples and Analogies

Quite often, people who are writing algorithms for the first time, tend to skip small steps assuming that they will happen automatically. One very common error is to skip the display statement, to display values that have been calculated. Some students write the complete logic to accept a set of values and perform elaborate calculations, but skip the display statement. You can ask them how they would feel if the program to generate their pay checks calculated the monthly pay, the tax and other deductions but missed out on the instruction to print the salary amount on the check.

You could use the following analogy to illustrate the point.

I had a servant called Robert. I once told him to go to the shop down the road and buy a pack of biscuits. He left immediately. Many hours later there was no sign of him. I set out in search of him. I walked towards the shop and was about to ask the shop owner about Robert, when I saw him standing close by with a pack of biscuits in his hand. When I asked what was wrong, he told me that as instructed he had bought the pack of biscuits. As he did not have any further instructions, he just waited there. I had missed one important instruction — to return home and give me the biscuits. Obviously this is a fictitious example, but is very representative of what could happen if an instruction is missed.

FAQ

Q 1. Are flowcharts actually used by software developers?
A 1. Yes, they are. A flowchart is a graphic tool, it is easier to communicate the logic using a flowchart.

Q 2. What is the convention used for the flow lines in a flowchart?
A 2. Generally, the lines flow from top to bottom and left to right.

Additional Inputs

As the programming logic is introduced for the first time, spend time on explaining the symbols used in a flowchart.

When flowchart and its various symbols are introduced, discuss the various keywords associated with the input, process, and output. For example, get, accept, and input can be used for input of data.
## Just a Minute...

Match the following verbs with appropriate symbols:

Get, Write, Stop, Add, Start, Multiply, Read, Subtract, Divide, Display, Input, Output, If.

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Symbol</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><img src="image1" alt="Symbol" /></td>
<td>If</td>
</tr>
<tr>
<td>2.</td>
<td><img src="image2" alt="Symbol" /></td>
<td>Add, Multiply, Subtract, Divide</td>
</tr>
<tr>
<td>3.</td>
<td><img src="image3" alt="Symbol" /></td>
<td>Get, Read, Input</td>
</tr>
<tr>
<td>4.</td>
<td><img src="image4" alt="Symbol" /></td>
<td>Stop, Start</td>
</tr>
<tr>
<td>5.</td>
<td><img src="image5" alt="Symbol" /></td>
<td>Write, Display, Output</td>
</tr>
</tbody>
</table>

**Solution:**

1. If
2. Add, Multiply, Subtract, Divide
3. Get, Read, Input
4. Stop, Start
5. Write, Display, Output
1.P.1

Procedure of Admission in NIIT

1. Start
2. Goes to NIIT Center
3. Fills the Enquiry Form
4. Appear for a Test
5. Get Counseled
6. Fills the Registration Form
7. Pays the Fee
8. Joins the Course and Gets the Student Kit
9. Stop

1.P.2
The correct steps are:
1. Start
2. Get the First Number
3. Get the Second Number
4. Add the Two Numbers
5. Display the Result
6. Stop
Sequence of Steps to Add Two Numbers

1. Start
2. Get the First Number
3. Get the Second Number
4. Add the Two Numbers
5. Display the Result
6. Stop

Display the Sum of Five Numbers
1.P.4

Display the Product of 2 and a Specified Number
Lesson Two

Experiences

While introducing relational operators, you need not introduce all the relational operators. Only introduce those that are required for the example given.

Since you already know programming languages, you might have a perception that the character datatype can store only one character. However as far as the PLT module is concerned, a variable of character datatype can store one or more characters.

The student may have difficulty in using the naming convention. The convention that is used is the hungarian notation. You may need to enforce it as it is one of the best practices followed by software developers.

You may need to explain the difference between datatypes and when to use which datatype.

Since the flowchart is a graphic tool, it is easier to communicate and document the logic, using flowcharts.

Additional Inputs

The faculty should explain the purpose of connectors given in the slides. These are not used in the student guide, as the flowchart comes in one page itself.

FAQ

Q 1. What do dotted lines in a flowchart indicate?

A 1. Dotted lines indicate that the flowchart is not complete but is a flowchart segment that has only the relevant steps depicted.

Solutions:

Just a Minute…

Identify the variable and constant data in the following situation:

Each day, the courier service delivers some letters. The number of letters is different each day. Regardless of the number of letters delivered by the courier service, they are paid a carrying charge of $5.

Variable:

Constant:

Solution:

Variable: Number of letters

Constant: Carrying charge $5
Identify the variables and constants from the list given below:

a) Age
b) Address
c) 21
d) “10, Kingsway Camp”
e) “Henri”
f) Name
g) “185”

Solution:
Constants: 21, “10, Kingsway Camp”, “Henri”, “185”

Variables: Age, Address, Name.

Draw a flowchart to accept item name, price, and quantity. You need to calculate value as the product of price and quantity, and display the calculated value and the item name using variables.
Solution:

2.P.1

a) OK
b) REJECT
c) GOOD

2.P.2

a) GOOD will be printed, since nX>nY and nX>100.
b) Nothing will be printed (even though nX is greater than nY, it is still less than 100).
c) Nothing will be printed (even though nY is greater than nX, it is still less than 100).
Find Out if a Number is Divisible by 5

1. Start
2. Accept nNumber
3. Is nNumber \( \% 5 = 0 \) ?
   - Yes: Display "Divisible by 5"
   - No: Display "Not Divisible by 5"
4. Stop
Display the Largest of the Three Numbers

Start

numeric nNum1, nNum2, nNum3, nLargest

Accept nNum1

Accept nNum 2

Accept nNum 3

Is nNum 1 ≥ nNum2 ?

Yes

nLargest = nNum1

No

nLargest = nNum2

Is nNum3 > nLargest ?

Yes

Display nNum3

No

Display nLargest

Stop
2.P.5

Check if the Age is Negative

Start

numeric nAge

Accept nAge

Is nAge = 0 ?

Yes

Display "Age cannot be zero"

No

Is nAge > 0 ?

Yes

Display nAge

No

Display "Age cannot be Negative"

Stop
Lesson Three

Experiences

You may need to spend time on explaining iteration as some students may find it difficult to grasp. You may need to spend time on explaining the concept of modules or procedures. You will also need to explain to the students the use of the symbol used for depicting procedures in flowcharts, since this is the first time they will be using this symbol.

Examples and Analogies

Since ‘procedures’ is a new concept, you will need to give some simple examples to understand this concept. I often use this example. I have a dog at home. Every morning the first thing I do is call out to the dog and ask it to fetch me the newspaper. So fetching the morning newspaper is a procedure that I can call everyday.

Additional Inputs

After the completion of an example, you should take the test case values and do a step by step dry run, so that the students are familiar with the dry run process and are clear on how to check the flow of control.
Solutions:

3.P.1

Display the Product of Ten Even Numbers

Start

numeric nNumber, nProduct, nEven

nNumber = 2, nEven = 0, nProduct = 1

nProduct = nNumber * nProduct

nNumber = nNumber + 2

nEven = nEven + 1

Is nEven < 10 ?

Yes

No

Display nProduct

Stop
3.P.2

Count the Number of Odd and Even Numbers

Start

numeric nNumber, nOdd, nEven, rCounter

nCounter = 1, nOdd = 0, nEven = 0

Accept nNumber

Is nNumber % 2 = 0 ?

Yes

nEven = nEven + 1

No

nOdd = nOdd + 1

rCounter = rCounter + 1

Is rCounter <= 50 ?

Yes

Display nOdd
Display nEven

No

Stop
3.P.3

Display the Highest of any Ten Numbers

Start

numeric nNumber, nLargest, nCounter

nCounter = 0, nLargest = 0

Accept nNumber

Is nNumber > nLargest?

No

Yes

nLargest = nNumber

nCounter = nCounter +1

Is nCounter < 10?

Yes

No

Display nLargest

Stop
3.P.4

**Modular Programming**

1. Start
2. Numeric nChoice
3. Display "1. Multiply 2. Divide"
4. Display "Enter Choice (1-2)"
5. Accept nChoice
6. If nChoice = 1 then
   - Multiply
7. If nChoice = 2 then
   - Divide
8. Stop
The names will be displayed infinite number of times, as the value of nCounter does not change. The value of nCounter will remain 0 and the condition will never become false. So the loop will never terminate. This is an infinite loop.

3.P.5

The names will be displayed infinite number of times, as the value of nCounter does not change. The value of nCounter will remain 0 and the condition will never become false. So the loop will never terminate. This is an infinite loop.
Lesson One

Solutions:

1. **Display Name and Age**

   - Start
   - Accept Student Name
   - Accept Student Age
   - Display Student Name, Student Age
   - Stop

2. **Display the Product of Two Numbers**

   - Start
   - Accept First Number
   - Accept Second Number
   - Multiply the Numbers
   - Display the Product
   - Stop
Lesson Two

Solutions:

1. Check if First Number is Divisible by the Second

```plaintext
Start

numeric nFirst, nSecond

Accept nFirst
Accept nSecond

Is nFirst % nSecond = 0?

Display "Divisible"

Display "Not Divisible"

Stop
```
2. Display the Day of the Week

Start

numeric rNum

Accept rNum

Is rNum=1 ? Yes Display "Sunday"

No

Is rNum=2 ? Yes Display "Monday"

No

Is rNum=7 ? Yes Display "Saturday"

No

Display "Enter a number between 1 and 7. Try again."

Stop
3. Calculate the Area

- Start
- \( \text{numeric } \pi = \frac{22}{7} \)
- \( \text{numeric } r, n\text{Area} \)
- Accept \( r \)
- \( n\text{Area} = \pi \times r \times r \)
- Display \( n\text{Area} \)
- Stop

4. Convert the Temperature from Fahrenheit to Celsius

- Start
- \( \text{numeric } n\text{Temp} \)
- \( \text{character } c\text{Scale} \)
- Accept \( n\text{Temp} \)
- Accept \( c\text{Scale} \)
- Is \( c\text{Scale} = "F" \)?
  - Yes: \( n\text{Temp} = (n\text{Temp} - 32) \times \frac{5}{9} \)
  - No: \( n\text{Temp} = \frac{(n\text{Temp} \times 9) + 32}{5} \)
- Display \( n\text{Temp} \)
- Stop
5. Display the Lowest of the Three Numbers

```
Start

numeric nNum1, nNum2, nNum3

Accept nNum1

Accept nNum2

Accept nNum3

Is nNum1 < nNum2 ?
  Yes     No nLowest = nNum2

nLowest = nNum1

Is nNum3 < nLowest ?
  Yes Display nLowest
  No

Display nNum3

Stop
```
Lesson Three

Solutions:

1. Display the Average of First Ten Odd Numbers

```
Start
numeric nNumber, nAverage, nOdd, nTotal

nNumber = 1, nOdd = 0, nTotal = 0

nTotal = nTotal + nNumber

nNumber = nNumber + 2

nOdd = nOdd + 1

Is nOdd < 10 ?

Yes

No

nAverage = nTotal / nOdd

Display nAverage

Stop
```
2. Display the Square of the First 100 Natural Numbers

- Start
- \textit{numeric} \textit{nNum}
- \textit{nNum} = 1
- \textit{Display nNum} \times \textit{nNum}
- \textit{nNum} = \textit{nNum} + 1
- \textit{Is nNum} \leq 100 ?
  - Yes
  - No
- \textit{Stop}
3.

**Display Fibonacci Series**

1. Start
2. Numeric nNum1, nNum2, nNum3
3. nNum1 = 0
4. nNum2 = 1
5. Display nNum1, nNum2
6. nNum3 = nNum1 + nNum2
7. Display nNum3
8. nNum1 = nNum2
9. nNum2 = nNum3
10. Is nNum2 = 89?
    - Yes
    - No
11. Stop
Display Total Number of Calls

Start

numeric nExtn1 = 0, nExtn2 = 2, nTotalCalls = 0

Is the call on extension 1?

Yes
  nExtn1 = nExtn1 + 1
  nExtn2 = nExtn2 + 1
  nTotalCalls = nTotalCalls + 1

No

Is nTotalCalls = 200?

Yes
  Display "Total number of calls on extension 1 are"
  Display nExtn1
  Display "Total number of calls on extension 2 are"
  Display nExtn2
  Stop

No
  nExtn2 = nExtn2 + 1
  nTotalCalls = nTotalCalls + 1
  Is the call on extension 1?
5. Calculate the Net Price of a Book

Start

numeri: nSNo = 1, nBookNo, nSP, nDiscount, nNetPrice
character cBookName, nCat

Accept nBookNo

Accept cBookName

Accept nSP

Accept nCat

Is cCat = '01'?

Yes

nDiscount = .2 * nSP

No

Is cCat = '02'?

Yes

nDiscount = .15 * nSP

No

Is cCat = '03'?

Yes

nDiscount = .1 * nSP

No

Is cCat = '04'?

Yes

nDiscount = .05 * nSP

No

Is cCat = '05'?

Yes

nDiscount = 0

No

Display "Categories can be as 01, 02, 03, 04, 05"

nNetPrice = nSP - nDiscount

A

B
6. Display the Status of the Stores Department

Start

- Stores
  - Character cStatus
  - Numeric nQty, nTotalQty

Accept cStatus
Accept nQty

- Is cStatus = "I"?
  - Yes: nTotalQty = nTotalQty + nQty
  - No: nTotalQty = nTotalQty - nQty

- Is cStatus = "O"?
  - Yes: Display "Wrong Status"
  - No: Display nTotalQty

Stop
7. Display the Sale Price

- Start
  - numeric nSalePrice, nDiscount, nDisSalePrice
  - Accept nSalePrice
  - Discount
    - nDiscount = nSalePrice * 0.05
    - nDisSalePrice = nSalePrice - nDiscount
  - Display nDisSalePrice
  - Stop

- Discount
  - Return
8. **Calculate the Average Marks of Students**

Start

numeric nCounter = 0

nCounter = nCounter + 1

Yes

Is nCounter < 20

No

Stop

CalculateAvg

numeric nMarks, nCtr = 0, nAverage, nTotal = 0

Accept nMarks

nTotal = nTotal + nMarks

nCtr = nCtr + 1

Yes

Is nCtr < 6?

No

nAverage = nTotal / 500

Display nAverage

Return
## SESSION PLAN: PROGRAMMING LOGIC AND TECHNIQUES

<table>
<thead>
<tr>
<th>Cycle #</th>
<th>Activity/Problem No.</th>
<th>Duration (In Mins)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cycle1</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OCR1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lesson 1: Objectives</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Input, Process, and Output</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Programs</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Problem-Solving Techniques</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>1.P.1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>1.P.2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>1.P.3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>1.P.4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td><strong>80</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Cycle2</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OCR1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lesson 2: Objectives</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Variables and Constants</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Data Types</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Using Operators</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Representing Decisions in a Flowchart</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>2.P.1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>2.P.2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>2.P.3</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2.P.4</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2.P.5</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td><strong>120</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Cycle3</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OCR1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lesson 3: Objectives</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Dry Run</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Iteration</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Modular Approach to Programming</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3.P.1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3.P.2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3.P.3</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3.P.4</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3.P.5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td><strong>120</strong></td>
<td></td>
</tr>
</tbody>
</table>