



<p>CL1002</p> <p><i>Programming Fundamentals Lab</i></p>	<p>Lab 01</p> <p>Introduction to Programming Fundamentals</p>
--	---

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

---

Fall 2024

## AIMS AND OBJECTIVES

---

The aim of this lab is to equip students with the foundational knowledge and **COMPUTATIONAL THINKING** necessary to understand and apply fundamental programming concepts.

By the end of the lab students will be able to: -

1. Apply problem-solving skills on real-world examples.
2. Make and use PAC (Problem Analysis Chart) methodology to break down a real-world problem.
3. Utilize IPO (Input-Process-Output) Charts.
4. Design and dry-run Flowcharts.

## INTRODUCTION

---

Programming is not just about coding in some computer language, although eventually you will end up doing it. Programming starts with understanding the problem first and designing a solution. This and the next few weeks are aimed at developing these fundamental skills first. Recognizing the actual problem and then designing and formalizing a solution.

This is intense and will require hard work from your end. Don't be afraid to make mistakes. We are here to correct them. Just do not give in to the temptation of getting the solution from somewhere else other than your brain. So, buckle up and fire up those sleeping neurons in your brain.

## PROBLEM SOLVING

---

Problem-solving in programming is: -

1. identifying a problem,
2. planning a solution, and
3. executing that solution to achieve the desired result.

It's fundamental in programming because it helps create efficient, effective, and scalable solutions.

### Example: Problem Solving 1

Imagine that you have the following image, which is a map of a road leading to the building shown in the picture.

- There is a car and trees.
- The car cannot cross the trees
- The road is divided into squares to calculate the steps of the car.
- Each square is considered as one step.
- The car can move omni-directionally.

**How can the car arrive at the building?**

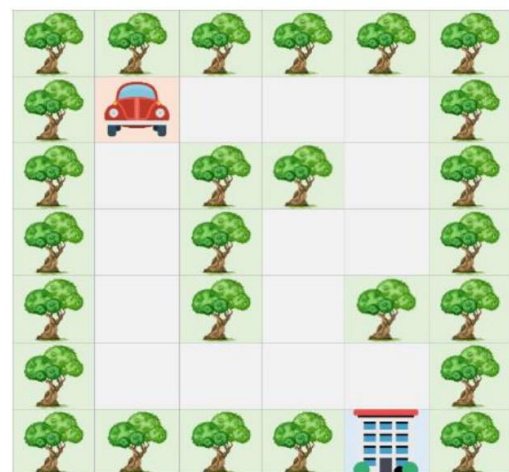


Figure 1(a) Problem Solving

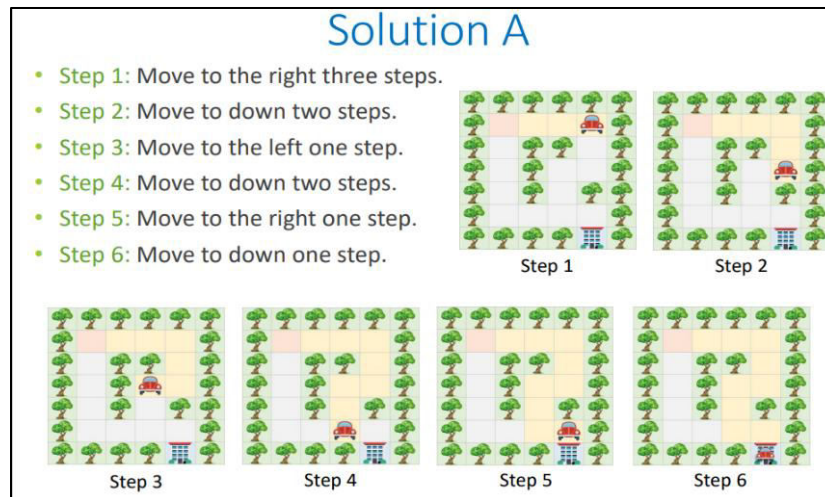


Figure: 1(b) Problem Solving

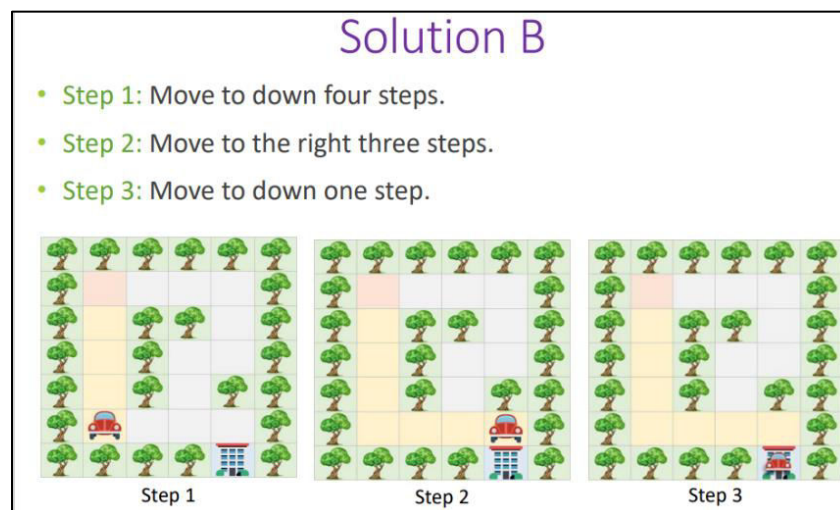


Figure: 1(c) Problem Solving

As we can see **Solution A** and **Solution B** are both correct solutions to the same problem. We as humans have an innate ability to see through our eyes and create a path. Now, imagine that the car is computer driven and the instructions you can send to the car are: -

1. Is it empty on your ([Left, Top, Right, Bottom])?  
(Computer replies yes or no)
2. Move to your ([Left, Top, Right, Bottom]).
3. Is destination at your ([Left, Top, Right, Bottom])?  
(Computer replies yes or no)

**Problem:** Can you write a set of instructions in plain English and using the above specific instructions to guide the car through the maze on, right?



Think about the following questions: -

1. Is it possible to come up with a set of instructions that will work on any map?
2. The instructions that you just gave will they work on any map?
3. Try using the instructions you just gave on the original map in figure 1(a). Do you end up in the destination?
4. If not give it another try?

### Example: Problem Solving 2

Consider yourself in a situation where you are a cashier (temporarily) at a department store. There are multiple denominations in the cash register with different quantities of them shown below. Your task is to take out exactly 75.43 USD



\$1	\$2	\$5	\$10	\$20	\$50	1¢	5¢	10¢	25¢	50¢
70 bills	1 bill	5 bills	3 bills	1 bill	0 bills	10 coins	5 coins	5 coins	2 coins	150 coins

Think about the following questions: -

1. Is there more than 1 correct answer?
2. What is the goal, what is the input?
3. What went in through your brain to solve the problem?
4. Can you write that process in form of steps in plain English?
5. Will these steps work for some other amounts let's say 89.23 USD?
6. If the amount of 75.43 USD and the quantities of bills and coins are not fixed is there a set of steps that can be taken to get the exact change every time?
7. Do these steps lead you to the correct answer for 75.43 USD?

### PROBLEM ANALYSIS CHART

The Problem Analysis Chart (PAC) is a tool used in troubleshooting problems, especially in programming and systems development. It allows problems to be broken down into smaller, more manageable parts and is often used to plan and document the steps needed to solve a problem using a programming language like C.

Given Data	Required Results
Section 1: Data given in the problem or provided by the user. These can be known values or general names for data, such as price, quantity, and so forth.	Section 2: Requirements for the output reports. This includes the information needed and the format required.
Processing Required	Solution Alternatives
Section 3: List of processing required. This includes equations or other types of processing, such as sorting, searching, and so forth.	Section 4: List of ideas for the solution of the problem.

Figure: 3 An Example of a Problem Analysis Chart (PAC)

In a Problem Analysis Chart there are 4 sections that we need to concentrate on. The Top Left section is the data for the problem we are currently trying to solve, so it will include things such as variables, numerical data, percentile and more. The Top Right section is the desired output we require which is what the problem is based on. The bottom left section is the processing section where the necessary steps are enlisted to ensure the correct output for the required results, and the bottom right section are alternate solutions as a problem can have multiple solutions to achieve the same result.

### Example: PAC

We have a problem in which we want to calculate the gross pay of an employee at the department store that you were previously working as a cashier (assume you are the general manager of that department store). Since you studied well you remember the formula for calculating the formula for gross pay. But I will write it down anyway for ease of use.

$$\begin{aligned} \text{Gross Pay} &= \text{Hours} * \text{Pay Rate} \\ \text{Payrate} &= 20\$/\text{HR.} \\ \text{Hours} &= 12 \end{aligned}$$

We need to develop a PAC chart for the above-mentioned problem.

Given Data	Required Results
Hours Pay Rate	Gross Pay
Processing Required	Solution Alternatives
$\text{GrossPay} = \text{Hours} * \text{PayRate}$	1. Define the hours worked and pay rate as constants. *2. Define the hours worked and pay rate as input values.

Figure: 4 Example of PAC chart

In the problem listed we identified that Hours and Pay Rate are the data that we are given so it is fed into the top left section of the chart. We require the gross pay of the employee so that goes to our required results section. A formula is required to calculate the gross pay so we can put it in the processing section, finally we can have multiple solutions to the problem as we can have an adaptive pay scale and hour scale.

Some more problems to solve



**Create PAC Charts of the following problems: -**

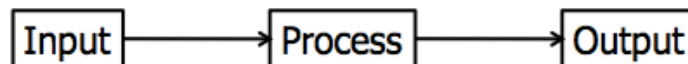
- Q1: Example 2 in the previous section. (denominations and exact cash).
- Q2: Find the largest of three numbers entered by the user. The user will take 3 random values as input. (Note they can be constants)
- Q3: Calculate the sum of the digits of a given number.
- Q4: Find whether a given number is even or odd. Alternatively, the user would take two numbers as input, multiply them and then determine if it is an Odd or Even number

## INPUT-PROCESS-OUTPUT

The IPO (Input-Process-Output) model is used in programming and system design for several important reasons. It provides a clear, structured way to conceptualize and develop software by breaking down the process into three fundamental steps: Input, Process, and Output.

IPO Chart shows:

- What data item are input.
- What processing takes place on that data.
- What information will be the result, the output.
- Where in the solution the processing takes place.



Example: IPO

Input	Processing	Module Reference	Output
Hours Worked Pay Rate	1. Enter Hours Worked 2. Enter Pay Rate 3. Calculate Pay 4. Print Pay 5. End	<i>Read</i> <i>Read</i> <i>Calc</i> <i>Print</i> <i>PayRollControl</i>	Gross pay

Figure: 5 An example IPO Chart of a Payroll System

In an IPO Chart we have 4 sections. Input, Processing, Module Reference and Output. Like the PAC chart we need to identify what are the input variables in our problem. The processing section has the same effect as the PAC chart but now in a more detailed layout where each step is enlisted. The Module Reference is one change from the PAC chart where each step of the processing section is converted into a command to which a computer will interpret. Such as Read, Calculate, Print, Loop, Write and Program control.

Some more problems to solve





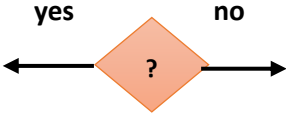
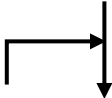


- Q5: Find whether a given number is even or odd. Alternatively, the user would take two numbers as input, multiply them and then determine if it is an Odd or Even number (This is the same problem as the above mentioned just create an IPO Chart of it).

## FLOWCHARTS

Flow charts are diagrams showing the exact sequence of logical steps. They use geometrical shapes and arrows to show processes, relationships, and data/process flow. In other words, flowcharts depict decisions and results of them.

In different fields, flowcharts are often used to analyze and manage processes. To put it simply, it helps you visualize what the processes look like. That way, you can see all the bottlenecks and flaws in them. The act of creating a chart is called flowchart.

	<b>Flowline:</b> Indicates the flow of logic by connecting symbols.
	<b>Terminals:</b> Represents the start and the end of a flowchart.
	<b>Input/Output:</b> Used for input & Output operation.
	<b>Processing:</b> Used for arithmetic operations and data-manipulations.
	<b>Decision:</b> Used for branching a flowline. Remember a flowline can never branch out by itself. Always use a decision box.
	<b>Looping:</b> Although flowlines can never branch out by themselves, they can always merge back in. Usually when they do it creates a loop. More on this later.



## Example: Flowchart

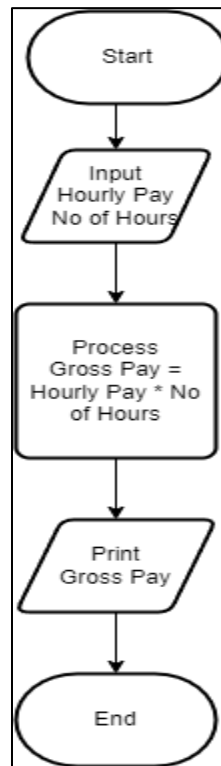


Figure: 6 Example of Flowchart

The start and End section of the Flow Chart represent the programs start and end boundaries. The Parallelogram where the Input and Print statements represent the Inputs and Outputs of the Program, and the Square Box represents the Processing required to achieve the desired output.

Some more problems to solve

**Create Flow charts of the following problems**

- Q6: Calculate the area of a rectangle given its length and width.  
Q7: •Convert a temperature from Celsius to Fahrenheit.  
Q8: •Calculate the average of three numbers.

## REFERENCES

---

- [https://csu.kau.edu.sa/Files/612009/Files/160348\\_Chapter0\\_CPIT110\\_v2.pdf](https://csu.kau.edu.sa/Files/612009/Files/160348_Chapter0_CPIT110_v2.pdf)
- <https://cs044.wordpress.com/wp-content/uploads/2017/02/chapter3-1.pdf>
- <https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial#:~:text=A%20flowchart%20is%20a%20diagram,easy%2Dto%2Dunderstand%20diagrams.>
- [https://harpercollege.pressbooks.pub/programmingfundamentals/chapter/input-process-output-model/#:~:text=The%20input%E2%80%93process%E2%80%93output%20\(structure%20for%20describing%20a%20process.](https://harpercollege.pressbooks.pub/programmingfundamentals/chapter/input-process-output-model/#:~:text=The%20input%E2%80%93process%E2%80%93output%20(structure%20for%20describing%20a%20process.)
- <https://www.studocu.com/row/document/inti-international-university/bahasa-kebangsaan/vbn-lecture-1-2-code/30284257>