# ASSIGNMENT 1

IPO CHARTS, PAC CHARTS, ALGORITHMS, PSEUDOCODE

Question 1:

**IPO Chart**

* **Input (what the program asks for):**

1.Previous money spent today

2.New transaction amount

3.Number of transactions in the last hour

4.Two usual countries for transactions

5.Current transaction country

* **Process (what the program does):**

1.Add today’s spending and the new transaction → check if it crosses the daily limit.

2.Compare the number of transactions in an hour against the safe limit (3).

3.Check if the transaction country matches either of the two usual ones.

* **Output :**

Alert messages or simply normal transaction messages.

**PAC Chart**

* **Problem:**  
  Banks need a simple way to spot unusual or suspicious transactions that might mean fraud, like spending too much in a day, shopping from strange locations, or making too many transactions too quickly.
* **Analysis:**  
  The program should ask the user about:

1. How much they’ve already spent today.
2. How much the new transaction is for.
3. How many transactions they’ve made in the last hour.
4. The usual countries where they shop.
5. The country where the current transaction is happening.

The system will then apply three checks:

* Daily spending limit (5000).
* No more than 3 transactions per hour.
* Must be in a familiar country.
* **Conclusion:**  
  By applying these rules, the program can flag suspicious activity and warn the user when something seems off.

**Algorithm**

1. Start the program.
2. Ask the user how much they already spent today.
3. Ask for the new transaction amount.
4. Add them up and check if the total goes above 5000.

If yes, show “ALERT: daily limit exceeded.”

If no, show “Within daily spending limit.”

1. Ask how many transactions happened in the last hour.

If more than 3, show “ALERT: too many transactions in one hour.”

Else, show “The hourly transaction count is normal.”

1. Ask for two usual countries and the current country of the transaction.

If the current country matches either usual one, show “"The Transaction has been processed”

Else, show “The Transaction location is unusual, ALERT”

1. End the program.

**Pseudocode**

START

Ask: "How much money was already spent today?"

Read prevMoney

Ask: "How much is this transaction?"

Read withdrawal

total = prevMoney + withdrawal

IF total > 5000 THEN

Print "ALERT: Daily spending limit exceeded"

ELSE

Print "Transaction is within daily limit"

Ask: "How many transactions in the last hour?"

Read transactionsInHour

IF transactionsInHour > 3 THEN

Print "ALERT: Too many transactions in one hour"

ELSE

Print "Transaction frequency is normal"

Ask: "Enter two usual countries for transactions"

Read country1, country2

Ask: "Enter the current transaction country"

Read presentCountry

IF presentCountry = country1 OR presentCountry = country2 THEN

Print "Transaction location is normal"

ELSE

Print "ALERT: Transaction location unusual"

STOP

Question 2:

**IPO Chart**

**1.Input:**

* + Two numbers: x and y.

**2.Process:**

* + Compare x and y against zero.
  + Decide if they match quadrant, axis, or origin.

**3.Output:**

* + A message showing where the point lies.

**PAC Chart**

**1.Problem:**  
We need a program that tells us where a point lies on the Cartesian plane — whether in one of the four quadrants, on the x-axis, on the y-axis, or at the origin.

**2.Analysis:**

* Input: two integers (x and y coordinates).
* Decision-making:
  + Quadrant I → x > 0, y > 0
  + Quadrant II → x < 0, y > 0
  + Quadrant III → x < 0, y < 0
  + Quadrant IV → x > 0, y < 0
  + On y-axis → x = 0, y ≠ 0
  + On x-axis → y = 0, x ≠ 0
  + At origin → x = 0, y = 0
* Output: A message showing the location.

**3.Conclusion:**  
By applying these conditions, we can always determine exactly where the point lies.

**Algorithm**

1. Start the program.
2. Ask the user to enter two numbers (x and y).
3. If x > 0 and y > 0, display “Point is in Quadrant I.”
4. Else if x < 0 and y > 0, display “Point is in Quadrant II.”
5. Else if x < 0 and y < 0, display “Point is in Quadrant III.”
6. Else if x > 0 and y < 0, display “Point is in Quadrant IV.”
7. Else if x = 0 and y ≠ 0, display “Point lies on y-axis.”
8. Else if y = 0 and x ≠ 0, display “Point lies on x-axis.”
9. Else if x = 0 and y = 0, display “Point lies at origin.”
10. End the program.

**Pseudocode**

START

Input x, y

IF x > 0 AND y > 0 THEN

Print "Point lies in Quadrant I"

ELSE IF x < 0 AND y > 0 THEN

Print "Point lies in Quadrant II"

ELSE IF x < 0 AND y < 0 THEN

Print "Point lies in Quadrant III"

ELSE IF x > 0 AND y < 0 THEN

Print "Point lies in Quadrant IV"

ELSE IF x = 0 AND y ≠ 0 THEN

Print "Point lies on Y-axis"

ELSE IF y = 0 AND x ≠ 0 THEN

Print "Point lies on X-axis"

ELSE

Print "Point lies at Origin"

STOP

Question 3:

**IPO chart**

**1.Input:**

* + Age (integer)
  + Eyesight test result (P/F)
  + Written test result (P/F)
  + Driving test result (P/F)
  + Medical certificate (Y/N, only if > 60)

**2.Process:**

Check if applicant is 18 or older.

If so, check eyesight.

If eyesight passed, check written test.

If written passed, check driving test.

If driving passed:

* + - If ≤ 60 → Eligible.
    - If > 60 → Check medical certificate.

**3.Output:**

* + Message stating **Eligible** or **Not Eligible** (with reason).

**PAC Chart**

**Problem:**  
Decide whether a person is eligible for a driving license based on age, eyesight exam, written test, driving test, and (if age > 60) medical certificate.

**Analysis:**

* If under 18, Not eligible.
* If eyesight failed, Not eligible (needs glasses).
* If written test failed, Not eligible (must retake).
* If driving test failed, Not eligible.
* If age ≤ 60 and all passed, Eligible.
* If age > 60 and has medical certificate, Eligible.
* If age > 60 and no certificate → Not eligible.

**Conclusion:**  
The program works step by step, checking each requirement. Eligibility is confirmed only if all conditions are satisfied.

**Algorithm**

1. Start
2. Input applicant’s age
3. If age < 18 → Print "Not eligible: underage", Stop
4. Input eyesight result (P/F)
5. If eyesight = F → Print "Need glasses", Stop
6. Input written test result (P/F)
7. If written = F → Print "Retake written test", Stop
8. Input driving test result (P/F)
9. If driving = F → Print "Not eligible: failed driving test", Stop
10. If age ≤ 60 → Print "Eligible for license", Stop
11. If age > 60 → Input medical certificate (Y/N)
12. If medical = N → Print "Not eligible: need medical certificate", Stop
13. Else Print "Eligible for license"
14. End

**Pseudocode**

START

Input age

IF age < 18 THEN

Print "Not eligible: underage"

STOP

ENDIF

Input eyesight

IF eyesight = F THEN

Print "You might need glasses"

STOP

ENDIF

Input written

IF written = F THEN

Print "You need to retake the written test"

STOP

ENDIF

Input driving

IF driving = F THEN

Print "Not eligible: failed driving test"

STOP

ENDIF

IF age <= 60 THEN

Print "Eligible for the license"

ELSE

Input medical certificate

IF medical = N THEN

Print "Not eligible: need medical certificate"

ELSE

Print "Eligible for the license"

ENDIF

ENDIF

STOP

Question 4:

**IPO Chart**

**1.Input:**  
5 card values between 1 and 13.

**2.Process:**

* 1. Accept 5 card values.
  2. Compare them using conditional checks.
  3. Look for the Full House pattern:
     + Three of one rank
     + Two of another rank
  4. If conditions are met → Full House. Otherwise → Not Full House.

**3.Output:**  
Display whether the poker hand is a Full House or not.

**PAC Chart**

**Problem:**  
Check if a poker hand of 5 cards (values between 1–13) forms a *full house* (three cards of one rank and two of another).

**Analysis:**

* Input: 5 integers between 1–13 (representing cards).
* Check different possible combinations of 3 matching cards and 2 matching cards.
* Use conditional statements to test the combinations.
* Output: State whether the hand is a *Full House* or *Not a Full House*.

**Conclusion:**  
The program compares cards systematically. If any condition satisfies the definition of a Full House, it prints “This hand is a full house*.”* Otherwise, it prints *“*This hand is not a full house.”

**Algorithm**

1. Start.
2. Input 5 integers a, b, c, d, e (each between 1–13).
3. Check if the cards satisfy the **Full House condition**:
   * Either **three cards of the same value** AND **two cards of the same value but different from the first three**.
   * This is done by comparing all possible combinations of 3 equal + 2 equal cards.
4. If the condition is true → Print **“It is a Full House”**.
5. Otherwise → Print **“Not a Full House”**.
6. End.

**Pseudocode**

BEGIN

DECLARE a, b, c, d, e AS INTEGER

DISPLAY "Enter 5 cards (1-13):"

READ a, b, c, d, e

IF ( (a=b AND a=c AND d=e) OR

(a=b AND a=d AND c=e) OR

(a=b AND a=e AND c=d) OR

(a=c AND a=d AND b=e) OR

(a=c AND a=e AND b=d) OR

(a=d AND a=e AND b=c) OR

(b=c AND b=d AND a=e) OR

(b=c AND b=e AND a=d) OR

(b=d AND b=e AND a=c) OR

(c=d AND c=e AND a=b) ) THEN

DISPLAY "It is a Full House"

ELSE

DISPLAY "Not a Full House"

ENDIF

END

Question 5:

**IPO Chart**

**1.Input:** User enters numbers repeatedly.

**2.Process:**

* 1. Use if else to check if input is between 0–9.
  2. If yes then increment the respective counter.
  3. If not then stop input and display counts.

**3.Output:** Display numbers with their occurrence count.

**PAC Chart**

**Problem:**  
Track how many times a user enters each one-digit number (0–9). Stop when a non-digit or invalid input is given and display counts.

**Analysis:**

* Each valid number should increment its own counter.
* Input ends when an invalid number is given.
* Display results in a table.

**Conclusion:**  
Use a loop with counters and conditional checks (if/else) to track occurrences.

**Algorithm**

Start.

Initialize counters c0, c1, …, c9 = 0.

Display instructions to the user.

Loop until input ends:

* Read number from user.
* If input is invalid → stop loop.
* Else if number is 0–9 → increment respective counter.
* Else → stop loop.

Display heading **Number | Occurrences**.

Print counts for digits 0–9.

End.

**5. Pseudocode**

START

SET c0 = 0, c1 = 0, c2 = 0, c3 = 0, c4 = 0

SET c5 = 0, c6 = 0, c7 = 0, c8 = 0, c9 = 0

PRINT "Enter digits (0–9). Enter any other number to stop."

LOOP FOREVER

PRINT "Enter number: "

READ num

IF num is not an integer THEN

BREAK

END IF

IF num == 0 THEN c0 = c0 + 1

ELSE IF num == 1 THEN c1 = c1 + 1

ELSE IF num == 2 THEN c2 = c2 + 1

ELSE IF num == 3 THEN c3 = c3 + 1

ELSE IF num == 4 THEN c4 = c4 + 1

ELSE IF num == 5 THEN c5 = c5 + 1

ELSE IF num == 6 THEN c6 = c6 + 1

ELSE IF num == 7 THEN c7 = c7 + 1

ELSE IF num == 8 THEN c8 = c8 + 1

ELSE IF num == 9 THEN c9 = c9 + 1

ELSE

BREAK

END IF

END LOOP

// Step 3: Display results

PRINT "Number Occurrences"

PRINT "0 ", c0

PRINT "1 ", c1

PRINT "2 ", c2

PRINT "3 ", c3

PRINT "4 ", c4

PRINT "5 ", c5

PRINT "6 ", c6

PRINT "7 ", c7

PRINT "8 ", c8

PRINT "9 ", c9

END