PF LAB 02

# TASK 1: PROCESSING A CUSTOMER’S ORDER AT A RESTAURANT

## ALGORITHM:

- START  
- HAVE THE WAITER TAKE THE ORDER

- ASK THE CUSTOMER IF HE HAS ANY SPECIAL ADD ONS/ REQUEST FOR HIS ORDER   
- PASS THE ORDER TO THE CHEF

- WAIT FOR THE ORDER TO COOK

- SERVE THE ORDER TO THE CUSTOMER

- PRESENT THE BILL TO THE CUSTOMER

- COLLECT THE BILL AND THANK THE CUSTOMER FOR VISITING

- END

**PSEUDO CODE:**

START

Input order

Initialize:

order\_item1,order\_item2,order\_item3,order\_item4

Initialize Bill

Bill = order\_item1 + order\_item2 + order\_item3 + order\_item4

If customer\_order is complete:

Print Bill

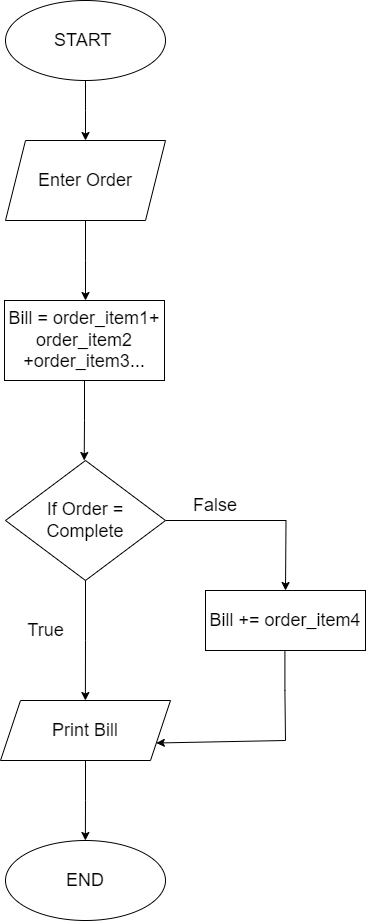
Else:

Bill += order\_item\_5

Print Bill

END

## FLOW CHART:

****

# TASK 2: ATM

## ALGORITHM:

1. START
2. Input Amount for Deposit
3. IF Deposited Amount is less than 1000: Display “Amount is too low”
4. ELSE: Add the deposited amount to the current balance
5. Check account validity
6. IF account is valid: Display current balance
7. ELSE: Display “Account invalid”
8. END

## PSEUDO CODE:

1. START
2. INT DEP\_AMOUNT
3. INT ACCOUNT\_VALIDITY
4. ACCOUNT\_VALIDITY = TRUE
5. INT CURRENT\_BALANCE
6. INPUT DEP\_AMOUNT
7. IF DEP\_AMOUNT <= 1000:
8. PRINT “AMOUNT IS TOO LOW “
9. ELSE: CURRENT\_BALANCE += DEP\_AMOUNT
10. IF ACCOUNT\_VALIDITY == TRUE:
11. PRINT CURRENT\_BALANCE
12. ELSE:
13. PRINT “ACCOUNT IS INVALID”
14. END

# TASK 3: WHICH NUMBER IS GREATEST

## ALGORITHM:

1. START
2. INPUT NUM1
3. INPUT NUM2
4. INPUT NUM3
5. INT GREATER\_NUMBER
6. INT GREATEST\_NUMBER
7. COMPARE NUM1 AND NUM2 AND ASSIGN GREATER TO GREATER\_NUMBER
8. COMPARE GREATER\_NUMBER TO NUM3 AND ASSIGN TO GREATEST\_NUMBER
9. PRINT GREATEST\_NUMBER

## PSEUDO CODE:

1. Int num1,num2,num3,greater\_num,
   1. greatest\_num
2. Input num1,num2,num3
3. If num1>num2: greater\_num = num1
4. Else: greater\_num = num2
5. If num2>num3: greatest\_num = num2
6. Else : greatest\_num = num3
7. Print greatest\_num

# TASK 4:

## ALGORITHM:

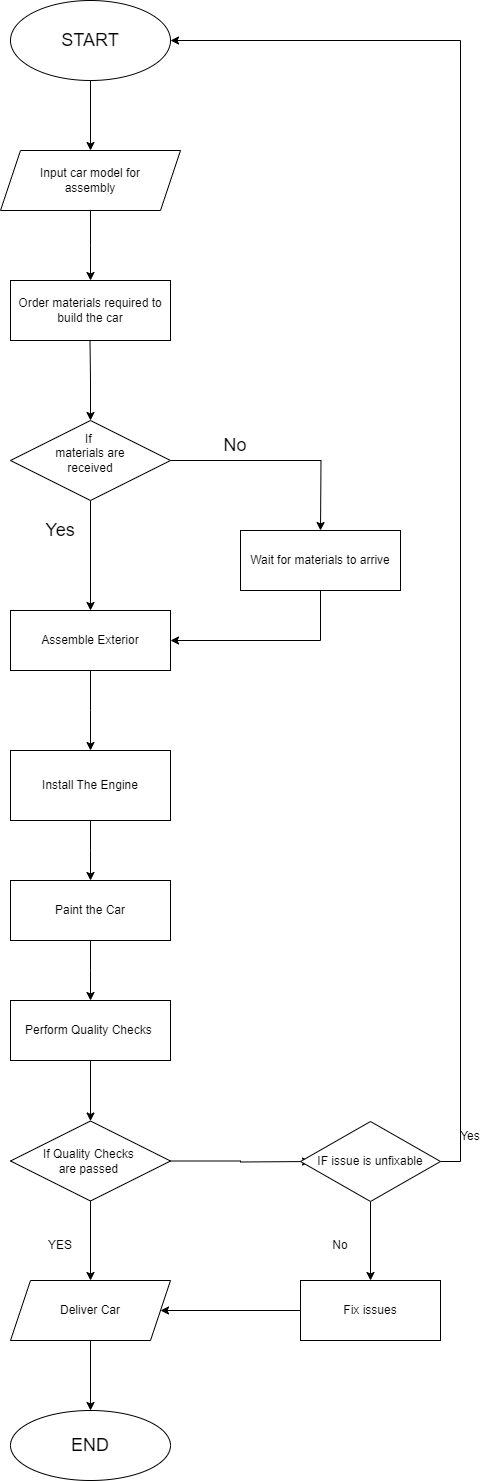
1. INPUT NUMBER BETWEEN 1 AND 12
2. 1=JANUARY,2=FEBRUARY,
3. 3=MARCH,4=APRIL,5=MAY,6=JUNE,
4. 7=JULY,8=AUGUST,9=SEPTEMBER,
5. 10=OCTOBER,11=NOVEMBER,
6. 12=DECEMBER
7. PRINT VALUE OF NUMBER

# TASK 5: A SMALL CALCULATOR

## PSEUDO CODE:

1. INT NUM1
2. INT NUM2
3. INT OPERATOR
4. INT RESULT
5. INPUT NUM1
6. INPUT NUM2
7. INPUT OPERATOR
8. IF OPERATOR == ‘+’:
9. RESULT = NUM1 + NUM2
10. PRINT RESULT
11. ELSE IF OPERATOR == ‘-‘:
    1. RESULT = NUM1 – NUM2
    2. PRINT RESULT
12. ELSE:
    1. PRINT”INVALID OPERATOR”

# Task 06: Assemble A Car



# TASK 07: Calculator

## Algorithm:

* START
* Enter First Number
* Enter Second Number
* Choose an Operator between + , - , \* , / , %
* If Operator is + :
  + Add First and Second Number and assign to Third Number
  + Print Third Number
* Else If Operator is - :
  + Subtract First Number From Second Number and assign to Third Number
  + Print Third Number
* If Operator is \* :
  + Multiply First and Second Number and assign to Third Number
  + Print Third Number
* Else If Operator is / :
  + Divide First Number From Second Number and assign to Third Number
  + Print Third Number
* Else If Operator is % :
  + Take Modulus of First and Second Number and assign to Third Number
  + Print Third Number
* END

# TASK 09: Why do we use .gitignore

We use .gitignore to make sure that some files are not committed to our github repo in order to prevent sensitive and confidential information like passwords from leaking to unwanted users. It is also useful to avoid committing unnecessary bloatware to our github repository

# TASK 10: Difference between Pseudo Code and Algorithm:

| Algorithm | Pseudo Code |
| --- | --- |
| Algorithm is a sequence of a  solution | It is a representation of the Algorithm |
| There are no rules to writing Algorithms | Certain Programming Rules and Fundamentals need to be followed while writing Pseudo Codes |
| Algorithms are more specific | Pseudo Codes are more general |
| It is a plan for solving a problem | It provides an overview for the solution’s logic. |