Task # 1:

Algorithm:

1. Greet the customer by saying **"Welcome, what would you like to eat ?"**

2. Display the **Menu**

3. Ask the customer to enter his **Order**

4. Read the **Order**

5. Ask for any **add on's**

6. Calculate **Bill**

7. If **add on included** then

8. Set **Total Bill** to **Bill + add on**

9. Else Set **Total Bill to Bill**

10. Get **Payment**

11. Display **Waiting Time**

Pseudo Code:

START

PRINT **"Welcome, what would you like to eat?"**

DISPLAY **Menu**

PRINT **"Enter your order?"**

INPUT **Order**

PRINT**"Any add on?"**

INPUT **add on**

IF **add on = Included**

THEN CALCULATE **Total Bill = Bill + add on**

ELSE

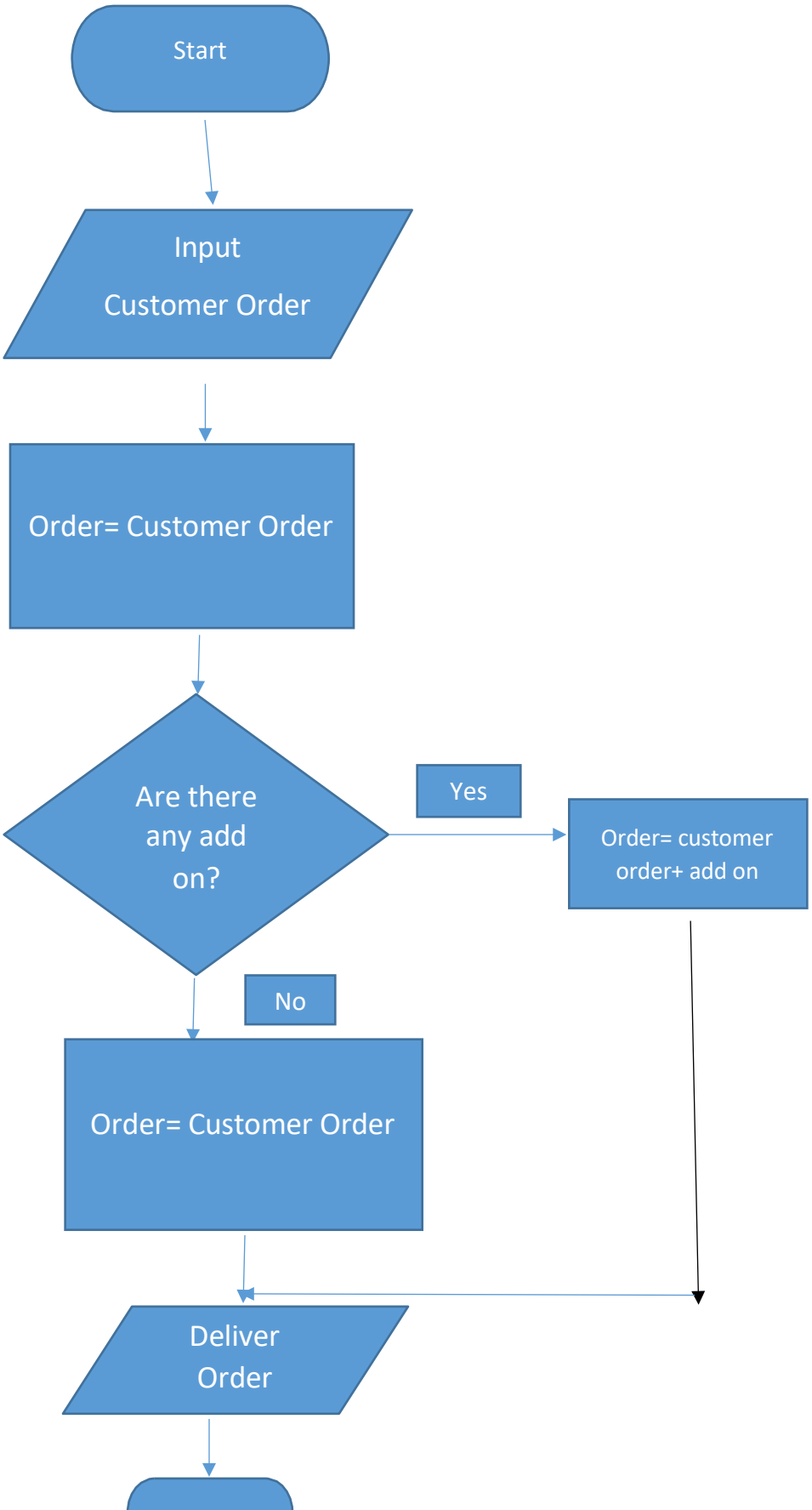
CALCULATE **Total Bill = Bill**

GET **Cash**

DISPLAY **ORDER TIME**

END

Flowchart:



Q#2.

Algorithm:

1. Ask the customer to enter the **Atm Card**
2. Read **Atm Card**
3. If **Atm Card** is Invalid then Display "Invalid account number"
4. Return **Atm**
5. Else Display **"How much amount you want to deposit"**
6. If amount exceeds Atm storage then display **"Insufficient balance"**
7. Else Give **Amount** and **Cash Receipt**

Pseudocode:

START
Input **Atm Card**
IF **Account**=**Invalid**
THEN print **"Invalid Account"**
RETURN **Atm Card**
ELSE
PRINT **"Enter Amount**"
IF **Amount**>**Acc_Balance**
THEN PRINT **"Insufficient Balance"**
ELSE
PRINT "**Amount** and **Receipt**"
END

Task# 3

Algorithm:

1. Input **num1** ,**num2**, **num3**
2. Compare **num1** and **num2**
3. If **num1** is greater than **num2** and **num3**
4. Display **num1** is largest
5. ELSE If **num2** is greater than **num1** and **num3**
6. Display **num2** is largest
7. ELSE If **num3** is greater than **num1** and **num2**
8. Display **num3** is largest

Pseudocode:

INPUT **Num1**, **Num2**, **Num3**

IF **Num1**>**Num2** and **Num1**>**Num3**

THEN PRINT **"Num1 is greatest"**

ELSE IF **Num2**>**Num3** and **Num2**>**Num1**

THEN PRINT **"Num2 is greatest**"

ELSE IF **Num3**>**Num1** and **Num3**>**Num2**

THEN PRINT **"Num3 is greatest"**

ENDIF

END

Task # 4

Algorithm:

1. Enter number from 1-12
2. IF number is less than 1 and greater than 12 then display "Incorrect number".
3. Else if **number==1** then display **January.**
4. Else if **number==2** then display **February**.
5. Else if **number==3** then display **March**.
6. Else if **number==4** then display **April.**
7. Else if **number==5** then display **May**.
8. Else if **number==6** then display **June.**
9. Else if **number==7** then display **July**.
10. Else if **number==8** then display **August.**
11. Else if **number==9** then display **September.**
12. Else if **number==10** then display **October**.
13. Else if **number==11** then display **November.**
14. Else if **number==12** then display **December**.

Q#5

Pseudocode:

Start

INPUT **num1**

INPUT **num2**

PRINT "enter operator **"+"** or **"-"**. "

IF operator== + THEN

RESULT= **num1**+**num2**

ELSE IF operator== "**-**"THEN

Result= **num1-num2**

ELSE

PRINT **"Invalid Operator please enter correct operator**"

ENDIF

END

Q#7

Algorithm:

1. Ask the user to enter **num_1** and **num**_2 and the operator (+,-,*,/)
2. Read **num_1 num_2** and operator
3. IF operator is + the Result is **num**_1 + **num_2**
4. ELSE IF operator is **-** the Result is **num_1** - **num_2**
5. ELSE IF operator is ***** the Result is **num_1** * **num_2**
6. ELSE IF **num_2** is not equal to 0 and operator is / then result is **num_1** / **num_2**
7. Else display **"Invalid operator"**
8. Display **Result**

Q#9

Why do we use .gitignore?

Exclude Unnecessary Files: Ignore files that are not needed in the repository, such as temporary files, build artifacts, and logs.

Protect Sensitive Information: Prevent sensitive files, like configuration files with passwords or API keys, from being tracked or shared.

Reduce Clutter: Keep the repository clean by avoiding the inclusion of files that are generated locally and not relevant to other contributors

Q#10

Difference between Algorithm and Pseudocode?

Algorithm	Pseudo Code
An algorithm is a systematic, logical approach that provides a step-by-step procedure for computers to solve a specific problem.	Pseudocode is a simplified version of programming codes, written in plain English language and used to outline a program before its implementation.
Algorithms can be expressed using flowcharts, natural language, and other methods.	Pseudocode includes various control structures such as repeat-until, if-then-else, while, for, and case.

Q#6

Flowchart: