**FLOWCHART:**

**Imagine you are automating the process of a vending machine. Create a flowchart that includes decision points for user input, selecting products, accepting payment, and dispensing the correct item. Include error-handling for invalid inputs and insufficient funds.**





.































**PSEUDO CODE:-**

**Write pseudocode to find the smallest number among three given variables. Implement a decision-making structure to compare the variables.**

1. Input 1 st number(num1)
2. Input 2 nd number(num2)
3. Input 3 rd number(num3)
4. If num1 &lt;num2 AND num1 &lt;num3 THEN
5. Print “num1 is smallest”
6. Else if num2 &lt;num1 AND num2&lt; num3 THEN
7. Print “num2 is smallest’
8. Else Print “num3 is smallest”

**Develop pseudocode for a basic calculator that performs multiplication and division. The pseudocode should prompt the user for two numbers and an operator, then display the result**

**of the operation.**

1. INPUT “Enter first number:”, num1
2. INPUT “Enter first number:”, num2
3. INPUT “Enter the arithmetic function you want to perform(\* or /):”, op
4. IF (op==\*) THEN Result=num1\*num2
5. ELSE Result=num1/num2
6. PRINT Result

**ALGORITHMS:-**

**Write an algorithm to determine whether a number is a prime number. The algorithm should iterate through possible divisors and determine if the number has any divisors other than 1 and itself.**

1. Start
2. Ask the user to input any number n
3. Now initialize the divisor d as d=2
4. Now we will iterate the number by dividing it with the possible divisors d.
5. Divide the number n by divisor d.If n%d==0 (i.e.,n is divisible by d), then n is not a prime number.
6. Increment the divisor by 1
7. Repeat step 4 till d==n
8. If the number has a divisor other than 1 and the number itself then print “This number is not a prime number”. Else print “This number is a prime number”.
9. Stop

**Q.Develop an algorithm for a program that takes two numbers as input and finds the Greatest Common Divisor (GCD) of the two numbers using the Euclidean algorithm.**

1. Start
2. Take two integers a and b as input
3. Divide the larger number by the smaller
4. If the larger number is completely divisible by the smaller number then the smaller number is the GCD of the two numbers. Go To step 6 else go to step 4.
5. If on division there exists any remainder then replace the larger number by the remainder and perform division again
6. Continue this process until one of the numbers becomes zero. The non-zero number at this stage will be the GCD of the original two numbers.
7. Print the GCD
8. Stop

**Create an algorithm that asks the user for a day number (1-365) and outputs the corresponding day of the week, assuming that January 1st is a Monday.**

1. Start
2. Ask the user to enter a day number(n) between 1 and 365.
3. Use the modulus operator and divide the day number by 7
4. If n%7==0 then print “Sunday” on screen
5. If n%7==1 then print “Monday” on screen
6. If n%7==2 then print “Tuesday” on screen
7. If n%7==3 then print “Wednesday” on screen
8. If n%7==4 then print “Thursday” on screen
9. If n%7==5 then print “Friday” on screen
10. If n%7==6 then print “saturday” on screen
11. Stop