**FLOW CHARTS**

Q1. You are working in a logistics company responsible for delivering packages. Design a flowchart to manage the process of receiving, sorting, and delivering packages. Include decision structures for handling fragile items and urgent deliveries.

Ans.

Start

Receive package

Check package details

Is package fragile

Label as Fragile

Use protective packaging

Continue on Next page

Handle with care

END

Deliver package

Normal sorting

Priority sorting

Load into vehicle

Is delivery urgent

2. 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.

Ans.

Start

Display product

Select Product

Is product selection valid

Invalid selection

Try again…….

Display Price

Continue on Next page

THANK YOU

Enter and Accept payment

No Sufficient payment

END

Dispense product and dispense change

(if necessary)

Sufficient funds transferred

**PSUEDO CODES**

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

Ans. Below is pseudocode based on the given question.

|  |
| --- |
| Start  // Step 1: Input three variables  Input num1, num2, num3  // Step 2: Declare a variable named smallest  Declare smallest  //Step 3: Check whether all number are equal  If num1 = num2 and num1 = num3  Print “All numbers are equal”    //Step 4: Check if num1 is smallest  Else if num1< num2 and num1<num3  Set smallest = num1  Print smallest  //Step 5: Check if num2 is smallest  Else if num2<num1 and num2<num3  Set smallest = num2  Print smallest  //Step 6: Otherwise num3 is smallest  Else  Set smallest = num3  Print num3  //Step 7: End the program  End |

Q2. 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.

Ans. Below is pseudocode based on the given question.

|  |
| --- |
| Start  // Step 1: Input two variables  Input num1, num2  // Step 2: chose an operator (\* or /)  Input operator  // Step 3: Declare a variable named result  Declare result  // Step 4: Perform the operation based on user input  If operator = ‘\*’  Set result = num1 \* num2  Print result  Else if operator = ‘/’  // Step 5: Check whether num2 is equal to zero (division by zero isn’t allowed)  If num2! = 0  Set result = num1/num2  Print result  else  Print “Division by zero is not allowed”  // Step 6: Handle invalid operator  Else  Print “Error: Invalid operator, please use \* for product and / or quotient”  // Step 7: End the program  End |

**ALGORITHMS**

1. 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 1and itself.

1. **Start:** Begin the algorithm
2. **Input:** Read the number ‘n’
3. **Check if ‘n’ is less than equal to 1:**
   * If ‘**n<=1**’ then, print “non-prime” and go to step7
4. **Set a flag:** Initialize “**is prime**” as ‘**true**’
5. Iterate through possible divisors from 2 to square root of ‘**n**’
   * For each number ‘**i**’ in this range, check if ‘**n**’ is divisible by ‘**i**’:
     1. If **n % i == 0** then;
        1. Set **is\_prime** to ‘**false**’
        2. Break the loop
6. **Check the flag:**
   * If **is\_prime** is still true, print "Prime"
   * Else, print "Not Prime"
7. End

2. 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. **Input:** read the day number ‘dayNumber’ between 1 and 365
3. **Check if dayNumber is within the valid range (1-365)**
   * If dayNumber < 1 or dayNumber > 365, print "Error: Invalid day number" and go to step 7 and end the program
4. **Calculate the day of the week**
   * Calculate the day of week by using the formula **‘**(**dayNumber-1**) **% 7’**
5. **Map the result to the corresponding day of week** 
   * 0 -> Monday
   * 1 -> Tuesday
   * 2 -> Wednesday
   * 3 -> Thursday
   * 4 -> Friday
   * 5 -> Saturday
   * 6 -> Sunday
6. **Print the day of the week**
7. **End**

3. 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.

Ans.

1. **Start**
2. **Input:** Read two numbers a and b
3. **Check if either number is zero:**
   * If **a == 0**, set **gcd** to b and go to step 7
   * If **b == 0**, set **gcd** to a and go to step 7
4. **Apply the Euclidean Algorithm:**
   * While **b** is not zero:
     1. Set **temp** to **b**
     2. Set **b** to **a % b**
     3. Set **a** to **temp**
5. **Set the GCD:**
   * When **b** becomes zero, a will contain the **GCD**. Set **gcd** to **a**
6. **Print the GCD**
7. **End**