**WEEK – 6**

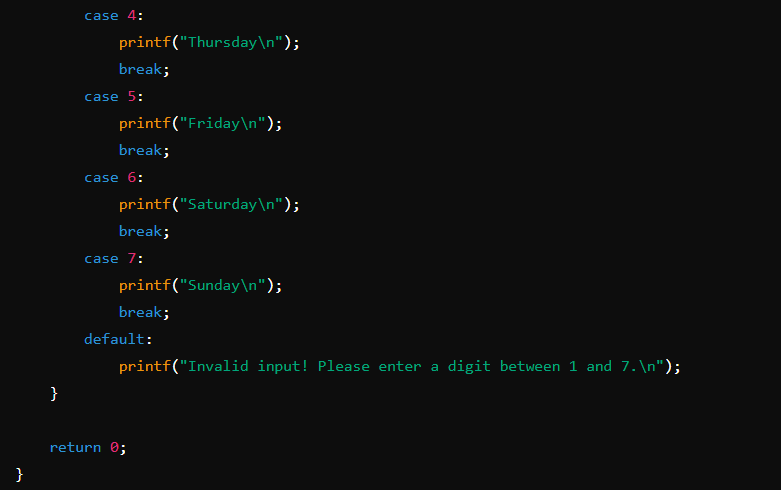
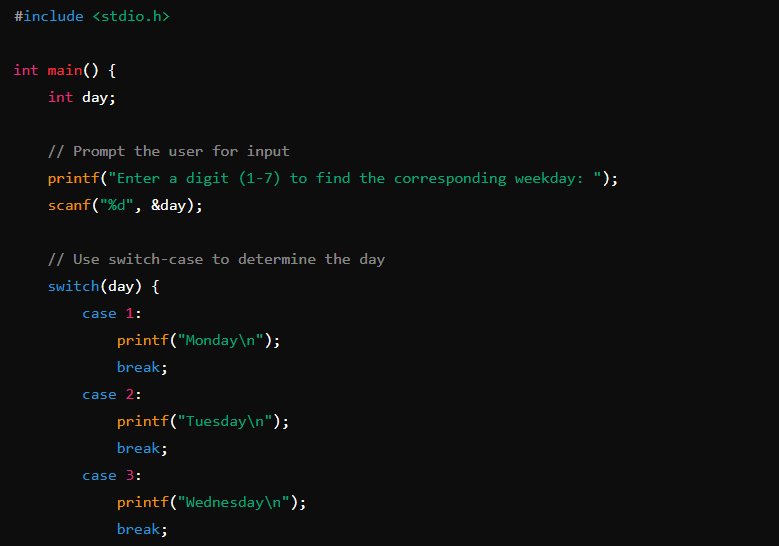
**1#** Develop a flow chart/algorithm and write an interactive program in C to find the week day by entering corresponding digit from the keyword (from 1 to 7) using switch case.

### **Step 1: Algorithm/Flow Chart Design**

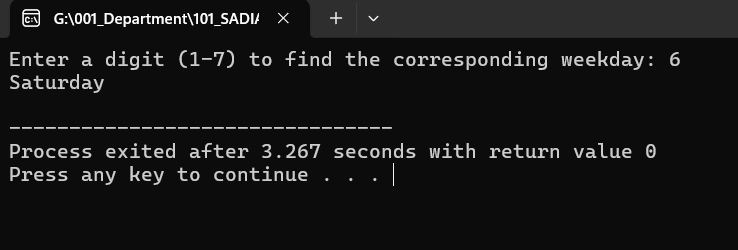
1. **Start**
2. **Input**: Ask the user to enter a digit (from 1 to 7).
3. **Process**:
   * Use a switch-case to match the input digit.
   * If the user enters:
     + 1: Print "Monday"
     + 2: Print "Tuesday"
     + 3: Print "Wednesday"
     + 4: Print "Thursday"
     + 5: Print "Friday"
     + 6: Print "Saturday"
     + 7: Print "Sunday"
   * If the input is out of range (not between 1 and 7), print an error message.
4. **End**: Program terminates.

### **Step 2: Flowchart**

1. Start
2. Prompt user for input (digit from 1-7)
3. Is input within range 1 to 7?
   * Yes -> Proceed to switch-case to match the weekday
   * No -> Print error message and exit
4. Print the corresponding weekday
5. End



OUTPUT: -



**2#** Develop a flow chart/algorithm and write a program to reverse and to sum of digits of a number which you entered from the keyboard.

### **Step 1: Algorithm/Flow Chart Design**

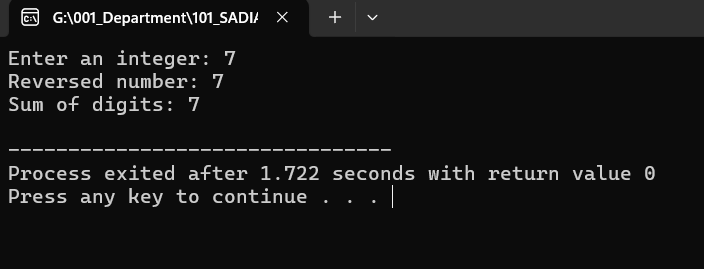
#### **Algorithm:**

1. **Start**
2. **Input**: Ask the user to enter a number.
3. **Initialize variables**:
   * sum = 0 (to store the sum of the digits)
   * reverse = 0 (to store the reversed number)
4. **Process**:
   * Use a loop to extract digits from the number.
   * In each iteration:
     + Extract the last digit using modulus (digit = number % 10).
     + Add the digit to sum.
     + Add the digit to reverse by multiplying the previous reverse by 10 and adding the extracted digit.
     + Remove the last digit from the number (number = number / 10).
5. **Output**: Print the reversed number and the sum of digits.
6. **End**

#### **Flowchart:**

1. **Start**
2. **Input**: Enter a number
3. Initialize sum = 0, reverse = 0
4. Loop until number > 0:
   * Extract the last digit (digit = number % 10)
   * Update sum = sum + digit
   * Update reverse = reverse \* 10 + digit
   * Remove the last digit (number = number / 10)
5. Print reverse and sum
6. **End**





**3#** Armstrong numbers are those numbers whose sum of cubes of each digit is equal to that number. For example: 153 = 13 + 33 + 53. Write a program to find all Armstrong Number in the range of 0 and 999.

**abc=a3+b3+c3**

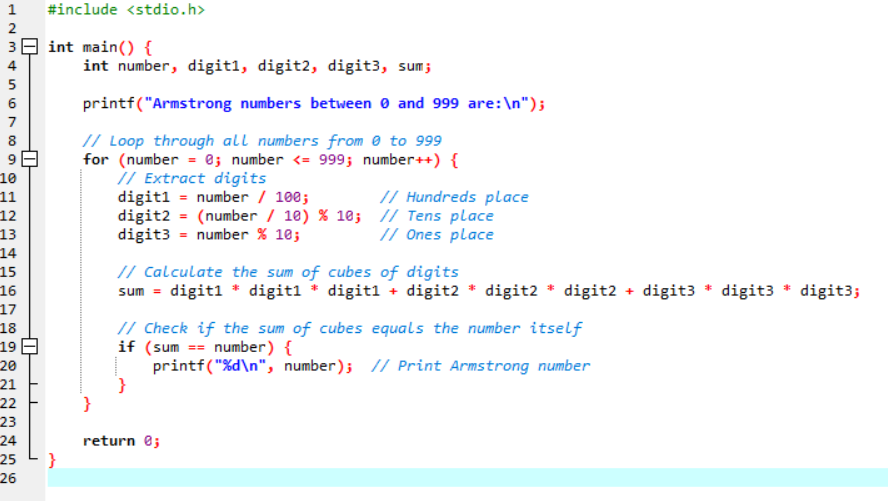
### **Step 1: Algorithm/Flow Chart Design**

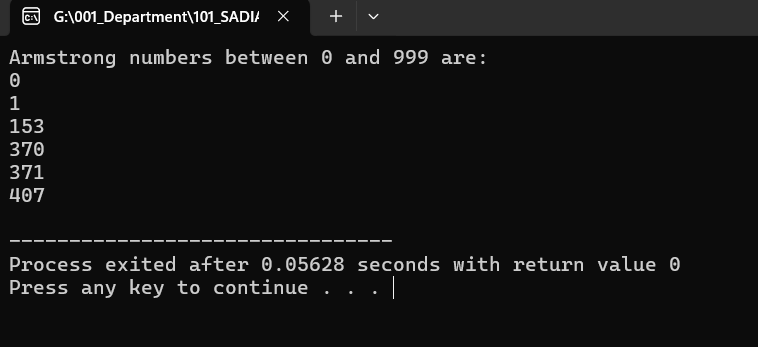
#### **Algorithm:**

1. **Start**
2. **Loop** through numbers from 0 to 999:
   * **For each number**:
     + Extract the hundreds, tens, and ones digits.
     + Compute the sum of the cubes of the digits.
     + Compare the sum to the original number.
     + If they are equal, print the number as an Armstrong number.
3. **End**

#### **Flowchart:**

1. **Start**
2. Loop through numbers from 0 to 999
3. For each number:
   * Extract the digits: hundreds, tens, ones.
   * Calculate the sum of the cubes of the digits.
   * If the sum equals the original number, print it.
4. **End**





**4#** Write a program to check the entered numbered is palindrome or not. Noted that palindrome means a number and its reverse number is same. For example: 1221.

A **palindrome** is a number (or a word) that reads the same forward and backward. For example, the number 1221 is a palindrome because its reverse is also 1221.

### **Step 1: Algorithm/Flow Chart Design**

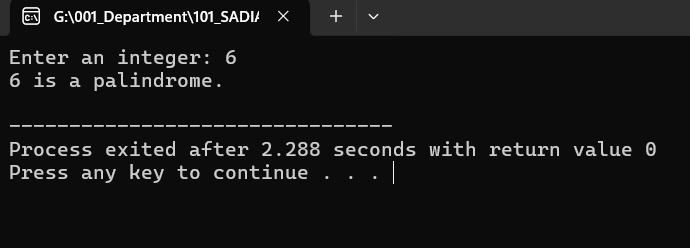
#### **Algorithm:**

1. **Start**
2. **Input**: Ask the user to enter a number.
3. **Initialize**:
   * originalNumber to store the input number.
   * reverse to 0 (to store the reversed number).
4. **Process**:
   * Use a loop to reverse the digits of the number:
     + Extract the last digit using modulus (digit = number % 10).
     + Update the reverse by multiplying the current reverse by 10 and adding the extracted digit.
     + Remove the last digit from the number using integer division (number = number / 10).
5. **Check**:
   * If originalNumber is equal to reverse, it is a palindrome.
   * Otherwise, it is not a palindrome.
6. **Output**: Print whether the number is a palindrome.
7. **End**

#### **Flowchart:**

1. **Start**
2. **Input**: Enter a number
3. Initialize originalNumber, reverse = 0
4. Loop until number > 0:
   * Extract last digit (digit = number % 10)
   * Update reverse = reverse \* 10 + digit
   * Remove last digit (number = number / 10)
5. Check if originalNumber == reverse
   * Yes -> Print "The number is a palindrome."
   * No -> Print "The number is not a palindrome."
6. **End**





**5#** Write a C program to print Fibonacci series: 0, 1, 1, 2, 3, 5, 8,13,……..

The **Fibonacci series** is a sequence of numbers where each number is the sum of the two preceding ones, starting from 0 and 1. The sequence goes: 0, 1, 1, 2, 3, 5, 8, 13, ...

The formula for the Fibonacci sequence is:

F(n)=F(n−1) + F(n−2)

with initial conditions:

F (0) = 0, F (1) = 1

### **Step-by-Step Explanation for the C Program:**

1. **Start** by printing the first two numbers of the series (0 and 1).
2. Use a loop to calculate the next numbers by adding the previous two numbers in the sequence.
3. Print the series up to the desired number of terms

