Write a program that swaps the values of two variables without using a temporary variable

Table of Contents

Program in C
Program in Java

Program In C

C

```c
#include <stdio.h>

void swap(int *a, int *b) {
    *a = *a + *b;
    *b = *a - *b;
    *a = *a - *b;
}

int main() {
    int num1 = 10;
    int num2 = 20;

    printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

    // Swap the values of num1 and num2
    swap(&num1, &num2);

    printf("After swapping: num1 = %d, num2 = %d\n", num1, num2);

    return 0;
}
```

Explanation:

- In this program, the swap function takes two pointers to integers (a and b).
- The swapping is achieved without using a temporary variable by utilizing the properties of addition and subtraction.
- Here's how it works:
- Add *a and *b, and store the result in *a.
- Now *a contains the sum of the original values of *a and *b.
- Subtract the original value of *b from the updated *a, and store the result in *b.
- Now *b contains the original value of *a.
- Finally, subtract the original value of *b from the updated *a, and store the result in *a.
- Now *a contains the original value of *b, effectively swapping the values.

In the main function, we demonstrate the usage of the swap function by swapping the values of two integers (num1 and num2).

Output:

C

```
Before swapping: num1 = 10, num2 = 20
After swapping: num1 = 20, num2 = 10
```

Can we create swapping programmes without using temporary variables or pointers in C?

Swapping the values of two variables without using a temporary variable or pointers is not

possible in C. The use of a temporary variable or pointers is necessary to perform the swap operation. Pointers allow us to indirectly access and modify the values stored in variables. Without pointers, we cannot achieve the necessary indirection required to swap the values without a temporary variable.

Program In Java

Java

```java
public class SwapNumbers {
    public static void swap(int[] arr) {
        arr[0] = arr[0] + arr[1];
        arr[1] = arr[0] - arr[1];
        arr[0] = arr[0] - arr[1];
    }

    public static void main(String[] args) {
        int num1 = 10;
        int num2 = 20;

        System.out.println("Before swapping: num1 = " + num1 + ", num2 = " + num2);

        // Create an array to hold the values of num1 and num2
        int[] arr = {num1, num2};

        // Swap the values using the swap method
        swap(arr);

        // Retrieve the swapped values from the array
```

```
        num1 = arr[0];
        num2 = arr[1];

        System.out.println("After swapping: num1 = " + num1 + ", num2
= " + num2);
    }
}
```

Explanation:

1. public class SwapNumbers: This line declares a public class named SwapNumbers.
2. public static void swap(int[] arr): This is a static method named swap that takes an integer array arr as a parameter. The method performs the swap operation on the array elements.
3. arr[0] = arr[0] + arr[1];, arr[1] = arr[0] – arr[1];, and arr[0] = arr[0] – arr[1];: These lines perform the swap operation by using arithmetic operations without using a temporary variable, similar to the original C code.
4. public static void main(String[] args): This is the main method that serves as the entry point of the program.
5. int num1 = 10; and int num2 = 20;: These lines initialize the variables num1 and num2 with the values 10 and 20, respectively.
6. System.out.println(“Before swapping: num1 = ” + num1 + “, num2 = ” + num2);: Prints a message indicating the values before swapping.
7. int[] arr = {num1, num2};: Creates an integer array arr and assigns it the values of num1 and num2. This is done to simulate passing the variables by reference as in the original C code.
8. swap(arr);: Calls the swap method and passes the array arr as an argument to swap its elements.

9. num1 = arr[0]; and num2 = arr[1];: Retrieves the swapped values of num1 and num2 from the array arr.
10. System.out.println("After swapping: num1 = " + num1 + ", num2 = " + num2);: Prints a message indicating the values after swapping.

Output

```
Before swapping: num1 = 10, num2 = 20
After swapping: num1 = 20, num2 = 10
```

Related posts:

1. Write a program that swaps the values of two variables.
2. Converts a number into integer, float, and string
3. Program to convert string to uppercase or lowercase
4. Program to prints the numbers from 1 to 10.
5. C prgoram to convert inch to feet
6. C program to convert KM to CM
7. C program to convert meter to centimeter
8. C program to calculate remainder, difference, division, product
9. C program to use printf() without semicolon " ; "
10. C program to swap two numbers using 2 variables
11. C program to find nth term using Arithmetic progrssion
12. C program to find sum of first n even positive numbers
13. C program to calculate sum of first n even numbers
14. C program to find nth odd number
15. C program to find sum of first n odd positive numbers
16. C program to calculate perimeter and area of a rectangle

17. C program to calculate perimeter and area of a square
18. C program to calculate Perimeter and Area of Circle
19. Function in C Programming
20. C Programming Q & A
21. Main function in C Programming Q and A
22. Void main in C Programming
23. Variables Q and A in C Programming
24. Write a C Program to find the percentage of marks ?
25. Write a c program to find age of a person ?
26. Write a c program to get table of a number
27. What is Break statement in C Programming ?
28. Write a c program to generate all combinations of 1, 2 and 3 using for loop.
29. Write a C program to print all the prime numbers between 1 to 50.
30. Write a C program to get factorial of a number ?
31. What is user defined function in C programming ?
32. Difference between C and C++ Programming ?
33. Difference between C, C++ and Java Programming
34. C program addition of numbers using pointer
35. C Syntax
36. Comments in C
37. Variables in C
38. Data types in C
39. Format specifiers in C
40. Type Conversion in C
41. Constants in C
42. Operators in C
43. Pre and Post Increament Practice Problems

44. Pre and Post Increament
45. Array in C
46. C Introduction
47. C Get Started
48. C Pointers
49. C History
50. C Program Compiling and running
51. C While loop
52. C Do While Loop
53. C For loop
54. break and continue statement
55. Control Statements in C
56. C if-else ladder
57. C if statements
58. C 2-Dimensional array
59. C String library functions
60. C Functions
61. C Functions Categories
62. C Actual Arguments
63. Write a program that prints the message “Hello, World!”
64. Write a program that asks the user to enter two numbers, and then prints the sum of those two numbers.
65. Write a program that asks the user to enter a number and then determines whether the number is even or odd.
66. Write a program that asks the user to enter a number and then calculates and prints its factorial.
67. Write a program that asks the user to enter a number N and then prints the first N

numbers in the Fibonacci sequence

68. Program to find the length of the string
69. What is identifier expected error
70. Difference between static and non static methods in Java
71. C String Input
72. C Character input
73. C Programming Variables MCQ
74. Object & Classes
75. C Programming find the output MCQs