**// #include <stdio.h>**

**// int main(){**

**//     int const data1 =10;//read onmly variable**

**//     // data1 = 50;**

**//     printf("001 data1 = %d\n",data1);**

**//     int \*Ptr = &data1;**

**//     \*Ptr = 500;**

**//     printf("002 data1 = %d\n",data1);**

**//     return 0;**

**// }**

**// #include <stdio.h>**

**// int main(){**

**//     int data1 =10;**

**//     data1 = 50;**

**//     return 0;**

**// }**

**// #include <stdio.h>**

**// int main(){**

**//     int const data1 =10;//read onmly variable**

**//     // data1 = 50;**

**//     printf("001 data1 = %d\n",data1);**

**//     int \*Ptr = (int \*)&data1;**

**//     \*Ptr = 500;**

**//     printf("002 data1 = %d\n",data1);**

**//     return 0;**

**// }**

**// #include <stdio.h>**

**// int const data2 =40;**

**// int main(){**

**//     // int const data1 =10;//read onmly variable**

**//     // data1 = 50;**

**//     printf("001 data1 = %d\n",data2);**

**//     int \*Ptr = (int \*)&data2;**

**//     \*Ptr = 500;**

**//     printf("002 data1 = %d\n",data2);**

**//     return 0;**

**// }**

**//case3**

**// #include <stdio.h>**

**// int main(){**

**//    int a =10;**

**//    int b =20;**

**//    int const \*Ptr =&a;**

**//    printf("Address of a = %p\n",&a);**

**//    printf("001 a address of pf Ptr = %p\n",Ptr);**

**//     Ptr = &b;**

**//     printf("Address of b = %p\n",&b);**

**//     printf("002 a address of pf Ptr = %p\n",Ptr);**

**//    //\*Ptr =50;**

**//     return 0;**

**// }**

**//case 2**

**// #include <stdio.h>**

**// int main(){**

**//    int a =10;**

**//    int const \*Ptr =&a;**

**//    \*Ptr =50;**

**//     return 0;**

**// }**

**//case3**

**// #include <stdio.h>**

**// int main(){**

**//    int a =10;**

**//    int b =20;**

**//    int \*const Ptr =&a;**

**//    printf("001  a = %d\n",&a);**

**//    \*ptr =40;**

**//     printf("002 a = %d\n",a);**

**//     return 0;**

**// }**

**#include <stdio.h>**

**int main(){**

**int a =10;**

**int b =20;**

**int const \*const Ptr =&a;**

**Ptr = &b;**

**printf("001  a = %d\n",&a);**

**\*ptr =40;**

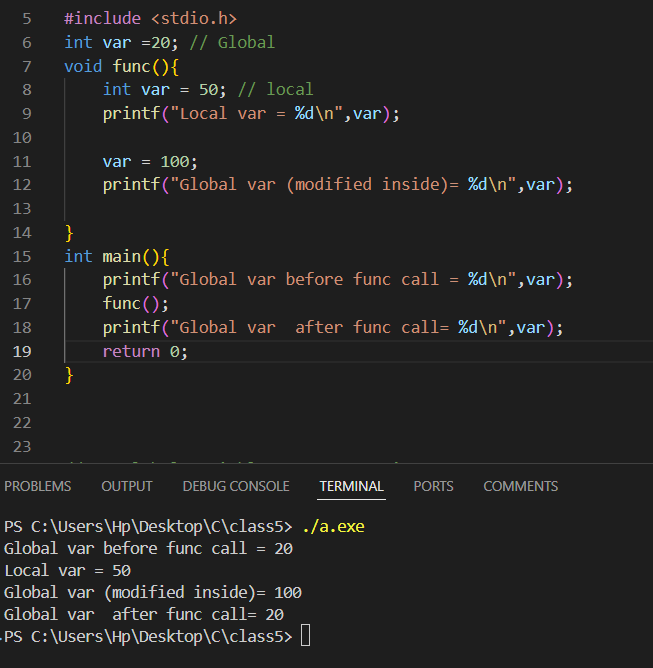
**printf("002 a = %d\n",a);**

**return 0;**

**}**

**1. Basic Global and Local Variable Usage**

* **Problem Statement**: Write a program that declares a global variable and a local variable with the same name. Modify and print both variables to demonstrate their scope and accessibility.



**2. Global Variable Across Functions**

* **Problem Statement**: Declare a global variable and create multiple functions to modify its value. Each function should perform a different operation (e.g., addition, subtraction) on the global variable and print its updated value.

A screen shot of a computer program

Description automatically generated

**3. Local Variable Initialization**

* **Problem Statement**: Write a program with a function that declares a local variable and initializes it to a specific value. Call the function multiple times and observe how the local variable behaves with each call.

A screen shot of a computer

Description automatically generated

**4. Combining Global and Local Variables**

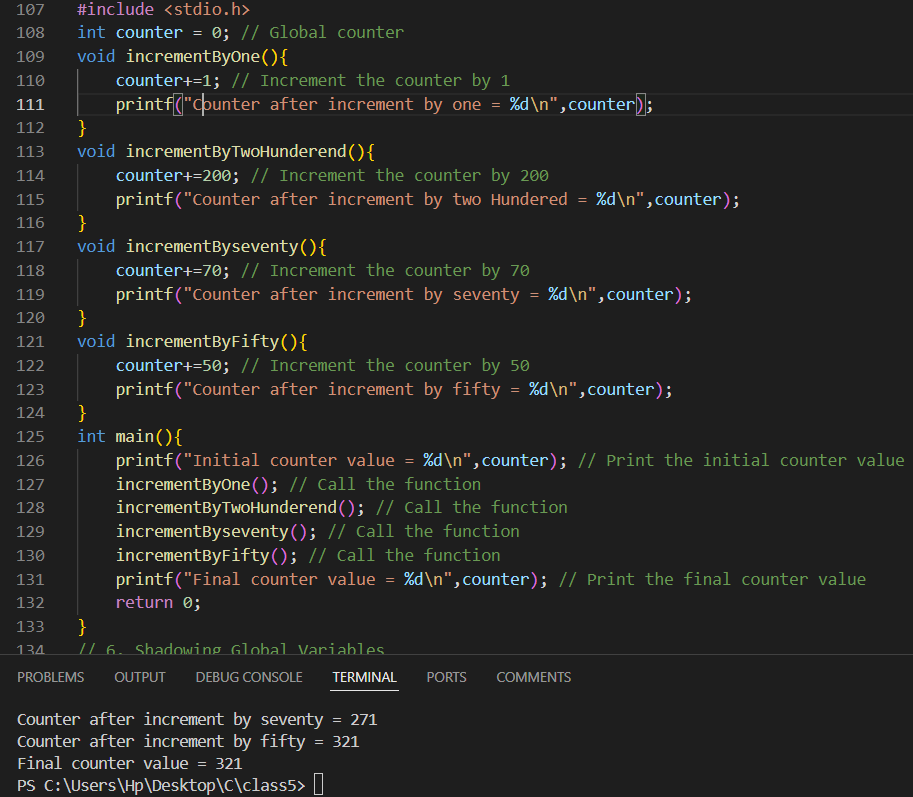
* **Problem Statement**: Write a program that calculates the sum of a global variable and a local variable inside a function. Print the result and explain the variable scope in comments.

A screen shot of a computer program

Description automatically generated

**5. Global Variable for Shared State**

* **Problem Statement**: Write a program that uses a global variable as a counter. Multiple functions should increment the counter and print its value. Demonstrate how global variables retain their state across function calls.



**6. Shadowing Global Variables**

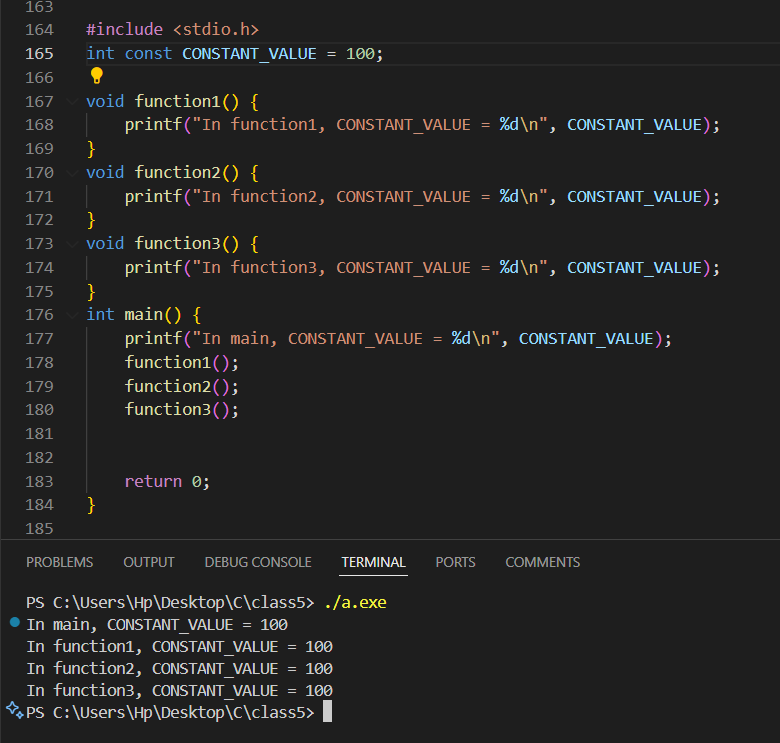
* **Problem Statement**: Write a program where a local variable in a function shadows a global variable with the same name. Use the global scope operator to access the global variable and print both values.

A screen shot of a computer program

Description automatically generated

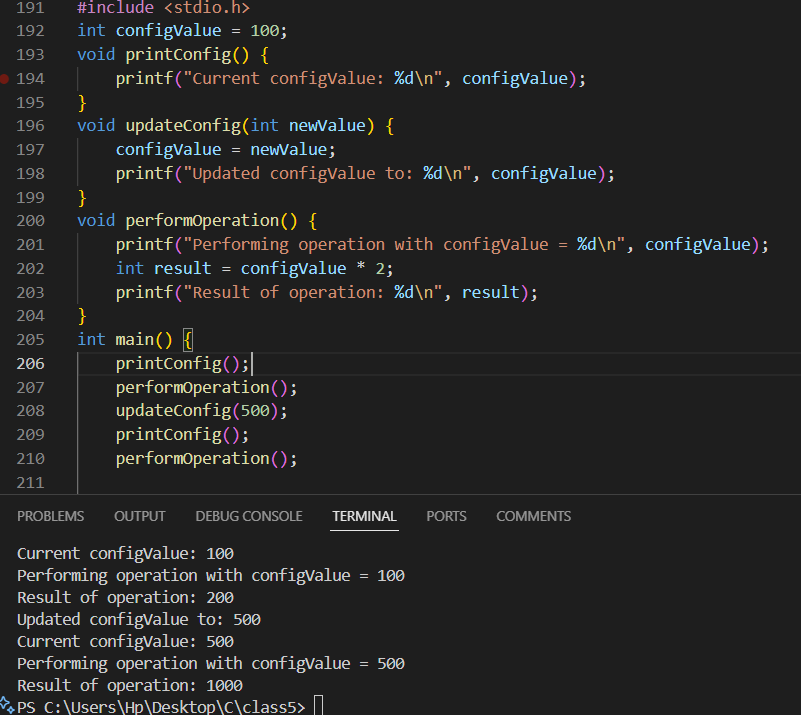
**7. Read-Only Global Variable**

* **Problem Statement**: Declare a global constant variable and write a program that uses it across multiple functions without modifying its value. Demonstrate the immutability of the global constant.



**8. Global Variable for Configuration**

* **Problem Statement**: Use a global variable to store configuration settings (e.g., int configValue = 100). Write multiple functions that use this global configuration variable to perform operations.



**9. Local Variables with Limited Scope**

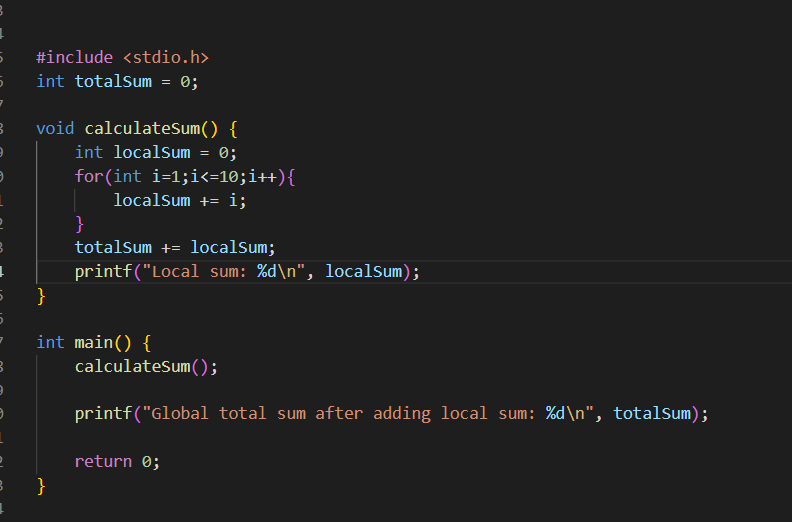
* **Problem Statement**: Write a program where local variables are declared inside a block (e.g., if or for block). Demonstrate that they are inaccessible outside the block.

A screen shot of a computer program

Description automatically generated

**10. Combining Local and Global Variables in Loops**

* **Problem Statement**: Write a program that uses a global variable to track the total sum and a local variable to store the sum of elements in an array. Use a loop to calculate the local sum, then add it to the global total.

A screen shot of a computer program

Description automatically generated

Problem statements on Static Storage classes

**1. Static Variable in a Loop**

* **Problem Statement**: Write a program that uses a static variable inside a loop to keep track of the cumulative sum of numbers from 1 to 10. The loop should run multiple times, and the variable should retain its value between iterations.

A screen shot of a computer program

Description automatically generated

**2. Static Variable to Count Iterations**

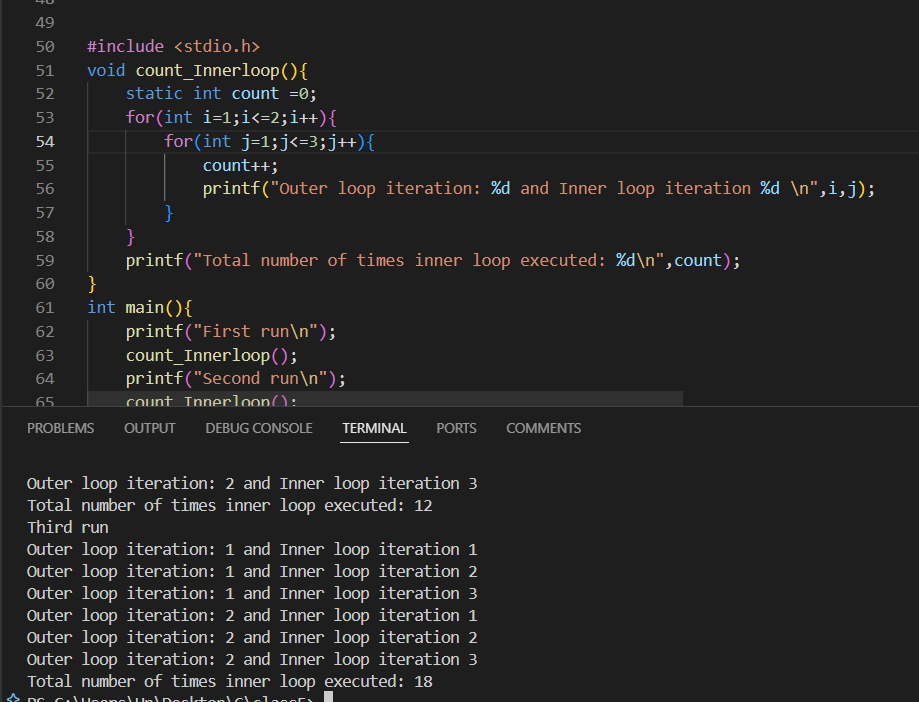
* **Problem Statement**: Use a static variable inside a loop to count the total number of iterations executed across multiple runs of the loop. Print the count after each run.

A screenshot of a computer program

Description automatically generated

**3. Static Variable in Nested Loops**

* **Problem Statement**: Use a static variable in a nested loop structure to count the total number of times the inner loop has executed across multiple runs of the program.



**4. Static Variable to Track Loop Exit Condition**

* **Problem Statement**: Write a program where a loop executes until a specific condition is met. Use a static variable to track and display the number of times the loop exited due to the condition being true.

A screenshot of a computer program

Description automatically generated

**5. Static Variable to Track Loop Re-entry**

* **Problem Statement**: Write a program where a static variable keeps track of how many times the loop is re-entered after being interrupted (e.g., using a break statement).

A screen shot of a computer

Description automatically generated

**6. Static Variable for Step Count in Loops**

* **Problem Statement**: Create a program with a loop that increments by a variable step size. Use a static variable to count and retain the total number of steps taken across multiple runs of the loop.

A screen shot of a computer program

Description automatically generated

Problem statement on const Type specifier

**1. Using const for Read-Only Array**

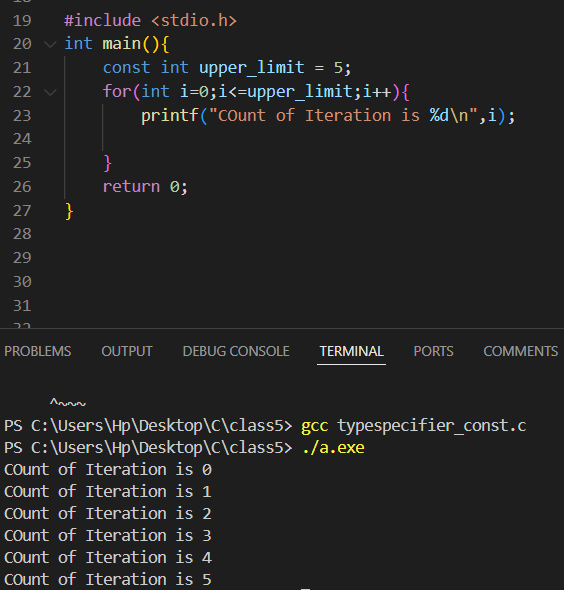
* **Problem Statement**: Declare an array of integers as const and use a loop to print each element of the array. Attempt to modify an element inside the loop and explain the result.

A screen shot of a computer program

Description automatically generated

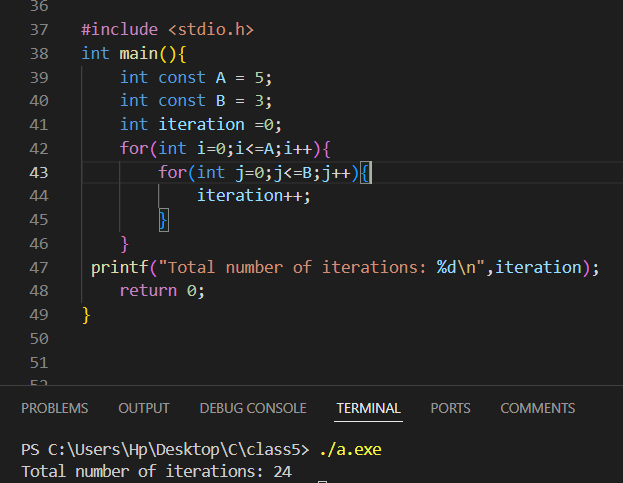
**2. const Variable as a Loop Limit**

* **Problem Statement**: Declare a const integer variable as the upper limit of a loop. Write a loop that runs from 0 to the value of the const variable and prints the iteration count.



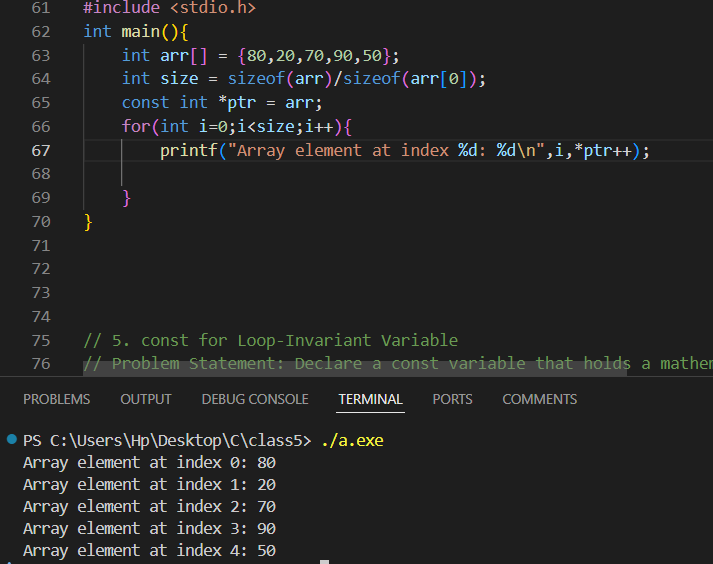
**3. Nested Loops with const Limits**

* **Problem Statement**: Use two const variables to define the limits of nested loops. Demonstrate how the values of the constants affect the total number of iterations.



**4. const for Read-Only Pointer in Loops**

* **Problem Statement**: Declare a const pointer to an integer and use it in a loop to traverse an array. Print each value the pointer points to.



**5. const for Loop-Invariant Variable**

* **Problem Statement**: Declare a const variable that holds a mathematical constant (e.g., PI = 3.14). Use this constant in a loop to calculate and print the areas of circles for a range of radii.

A screen shot of a computer

Description automatically generated

**6. const Variable in Conditional Loops**

* **Problem Statement**: Use a const variable as a termination condition for a while loop. The loop should terminate when the iteration count reaches the value of the const variable.

A screen shot of a computer program

Description automatically generated

**7. const and Immutable Loop Step Size**

* **Problem Statement**: Declare a const variable as the step size of a for loop. Use this step size to iterate through a range of numbers and print only every nth number.

A screen shot of a computer program

Description automatically generated

**8. const Variable for Nested Loop Patterns**

* **Problem Statement**: Use two const variables to define the number of rows and columns for printing a rectangular pattern using nested loops. The dimensions of the rectangle should be based on the const variables.

A screen shot of a computer program

Description automatically generated