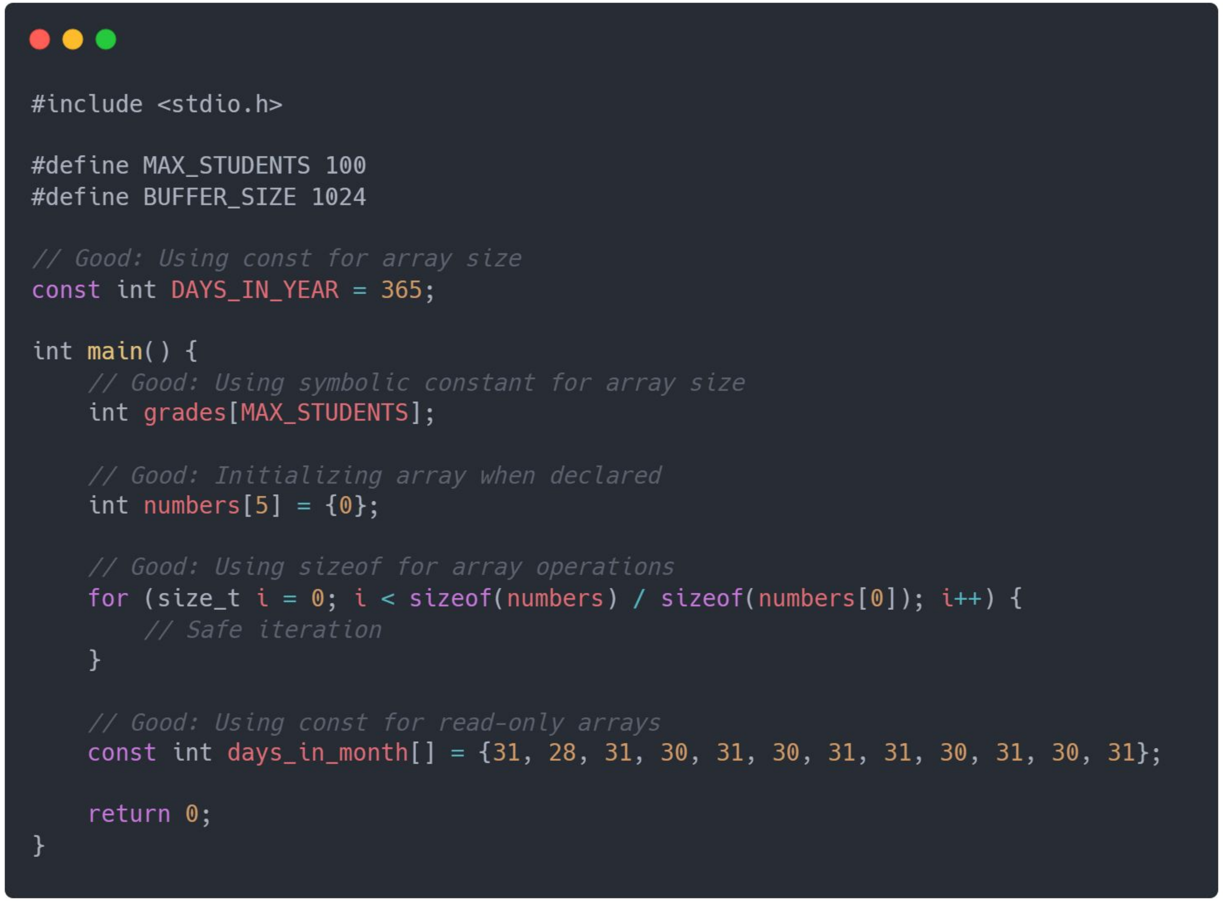
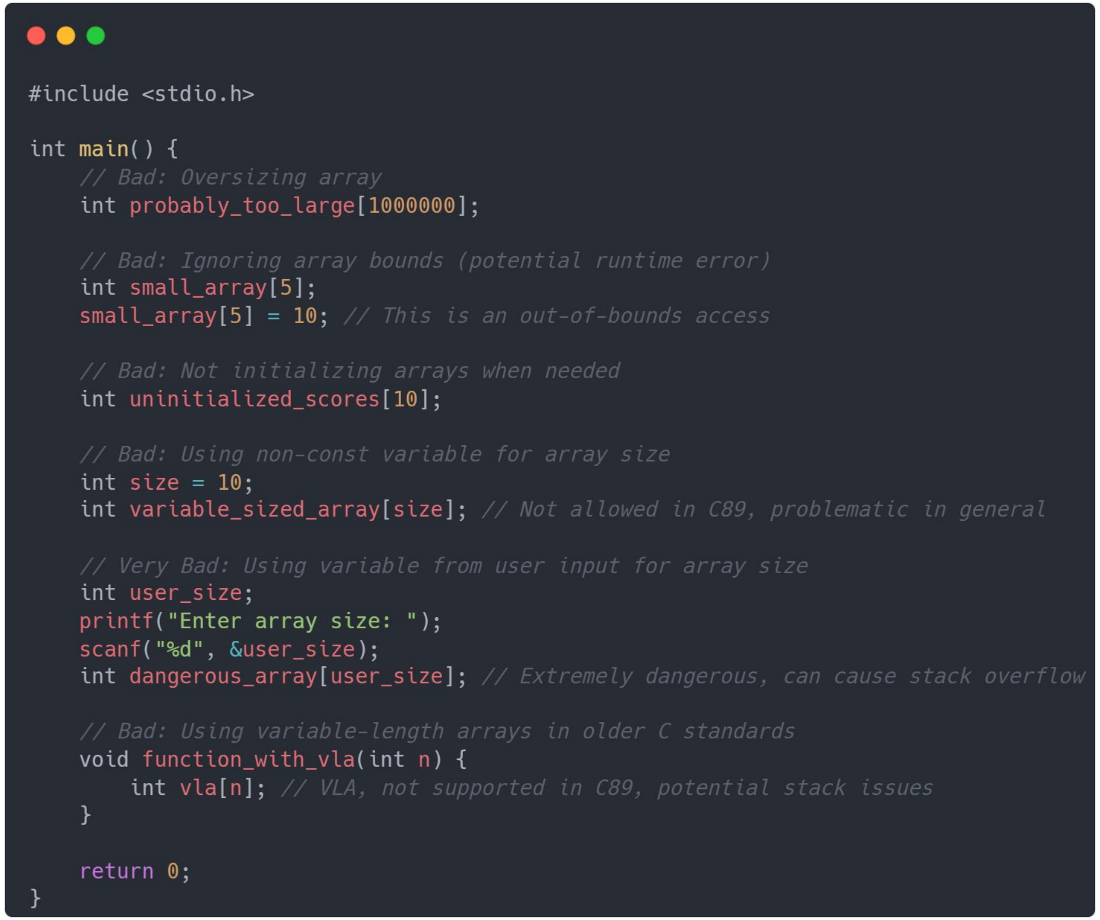
Postrehy tyzden 2

Statická alokácia



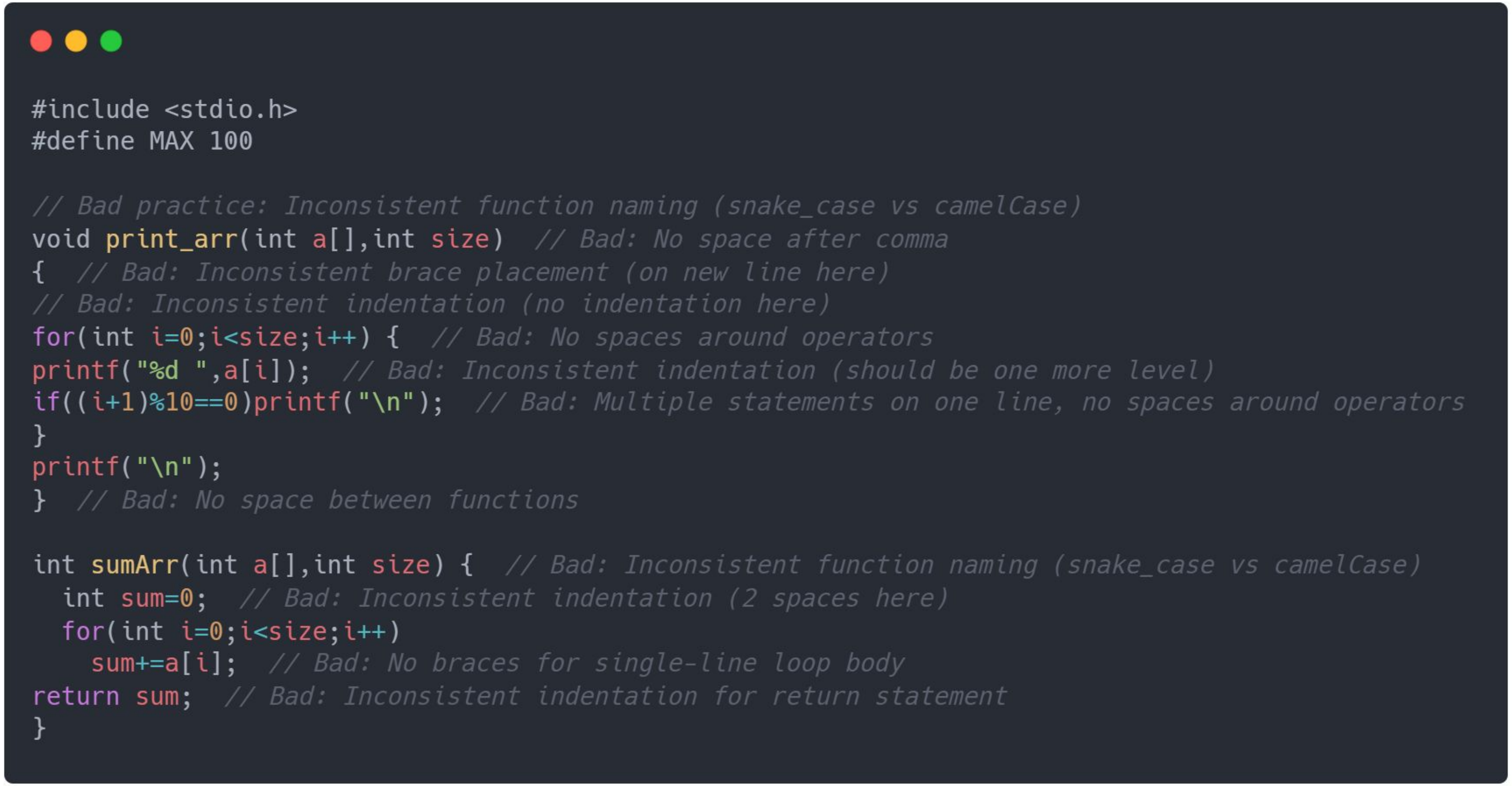
```c
#include <stdio.h>

#define MAX_STUDENTS 100
#define BUFFER_SIZE 1024

// Good: Using const for array size
const int DAYS_IN_YEAR = 365;

int main() {
    // Good: Using symbolic constant for array size
    int grades[MAX_STUDENTS];

    // Good: Initializing array when declared
    int numbers[5] = {0};

    // Good: Using sizeof for array operations
    for (size_t i = 0; i < sizeof(numbers) / sizeof(numbers[0]); i++) {
        // Safe iteration
    }

    // Good: Using const for read-only arrays
    const int days_in_month[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

    return 0;
}
```

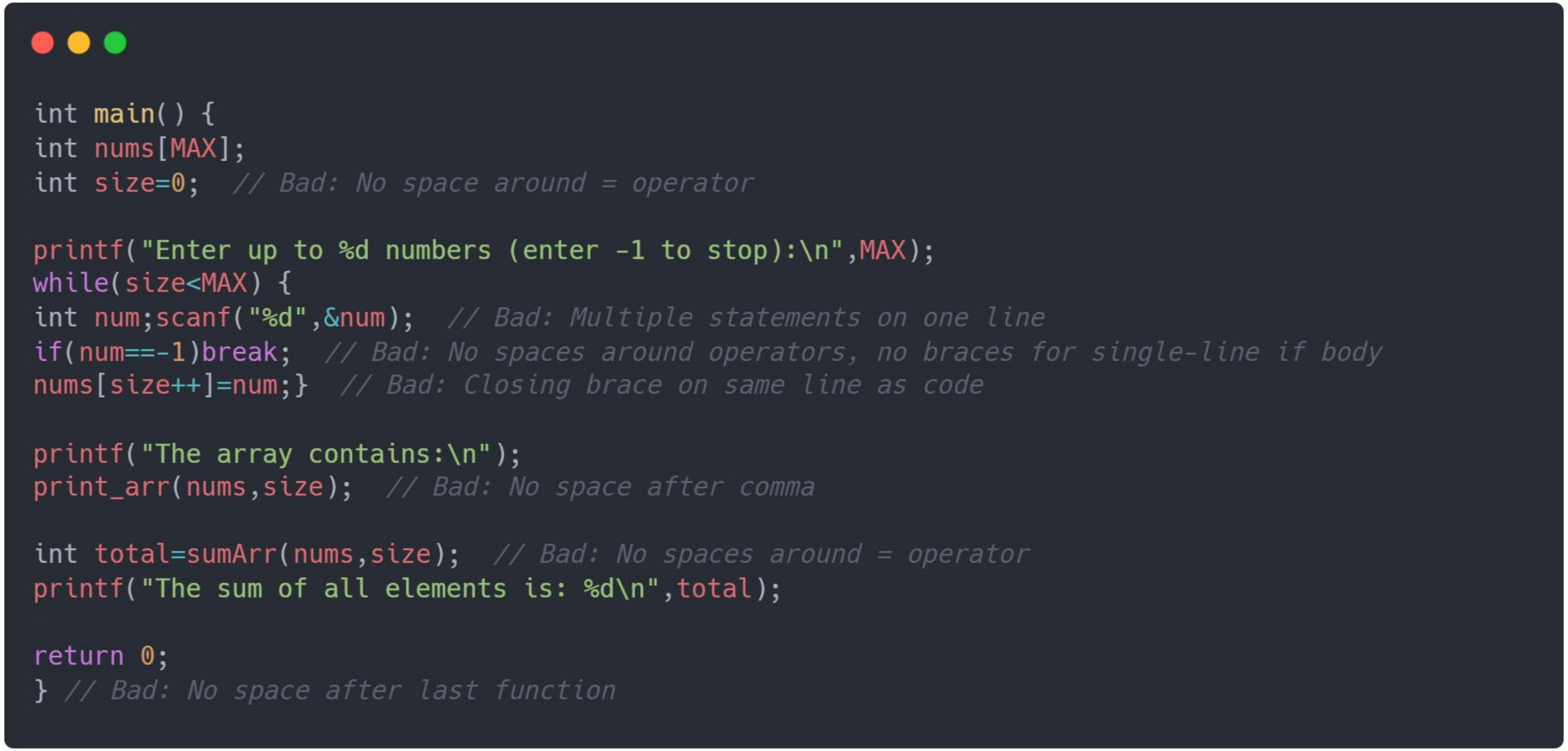
Statická alokácia



```c
#include <stdio.h>

int main() {
    // Bad: Oversizing array
    int probably_too_large[1000000];

    // Bad: Ignoring array bounds (potential runtime error)
    int small_array[5];
    small_array[5] = 10; // This is an out-of-bounds access

    // Bad: Not initializing arrays when needed
    int uninitialized_scores[10];

    // Bad: Using non-const variable for array size
    int size = 10;
    int variable_sized_array[size]; // Not allowed in C89, problematic in general

    // Very Bad: Using variable from user input for array size
    int user_size;
    printf("Enter array size: ");
    scanf("%d", &user_size);
    int dangerous_array[user_size]; // Extremely dangerous, can cause stack overflow

    // Bad: Using variable-length arrays in older C standards
    void function_with_vla(int n) {
        int vla[n]; // VLA, not supported in C89, potential stack issues
    }

    return 0;
}
```

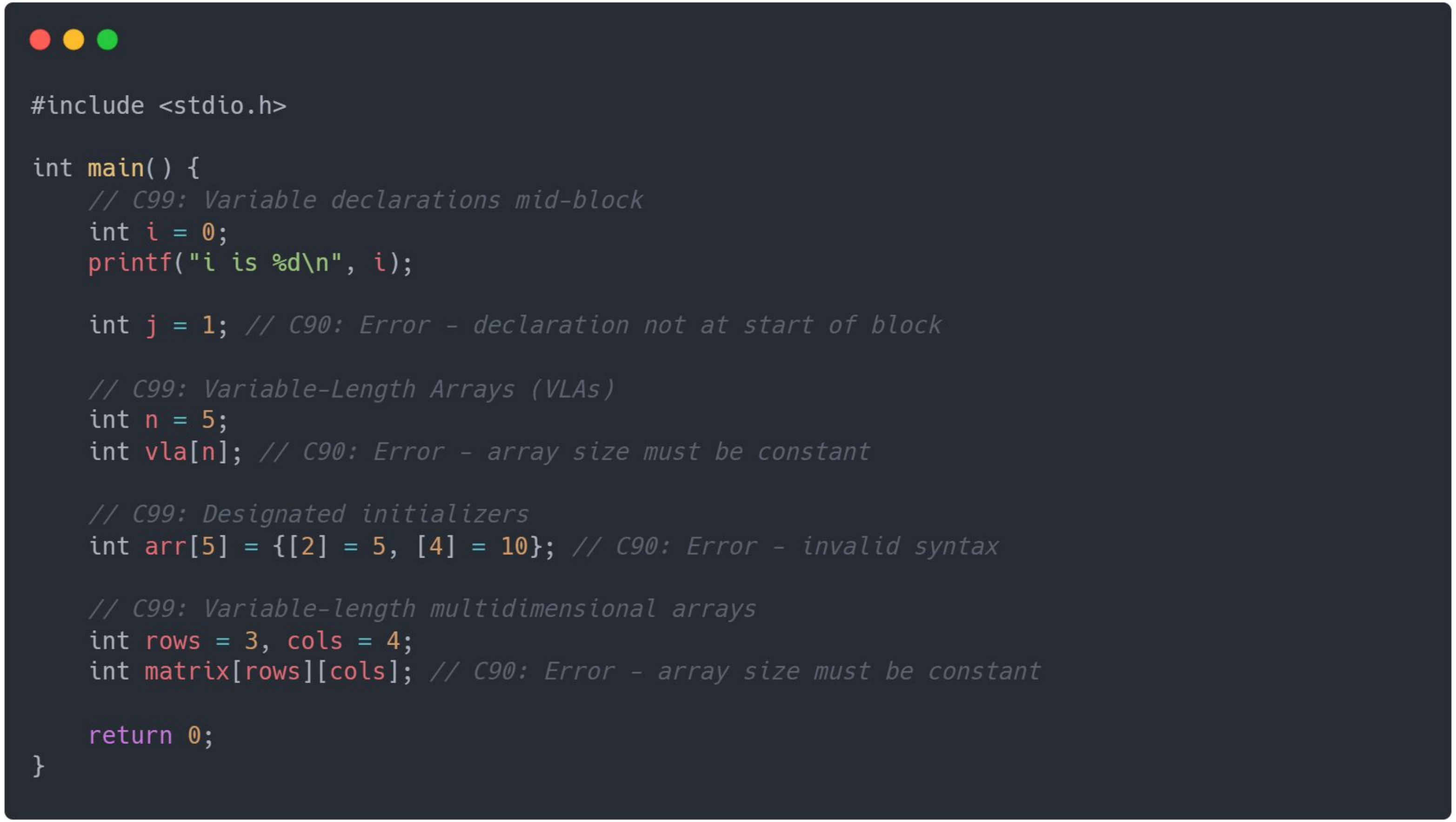
Formatovanie kodu


```c
#include <stdio.h>
#define MAX 100

// Bad practice: Inconsistent function naming (snake_case vs camelCase)
void print_arr(int a[],int size)  // Bad: No space after comma
{  // Bad: Inconsistent brace placement (on new line here)
// Bad: Inconsistent indentation (no indentation here)
for(int i=0;i<size;i++) {  // Bad: No spaces around operators
printf("%d ",a[i]);  // Bad: Inconsistent indentation (should be one more level)
if((i+1)%10==0)printf("\n");  // Bad: Multiple statements on one line, no spaces around operators
}
printf("\n");
}  // Bad: No space between functions

int sumArr(int a[],int size) {  // Bad: Inconsistent function naming (snake_case vs camelCase)
  int sum=0;  // Bad: Inconsistent indentation (2 spaces here)
  for(int i=0;i<size;i++)
    sum+=a[i];  // Bad: No braces for single-line loop body
return sum;  // Bad: Inconsistent indentation for return statement
}
```

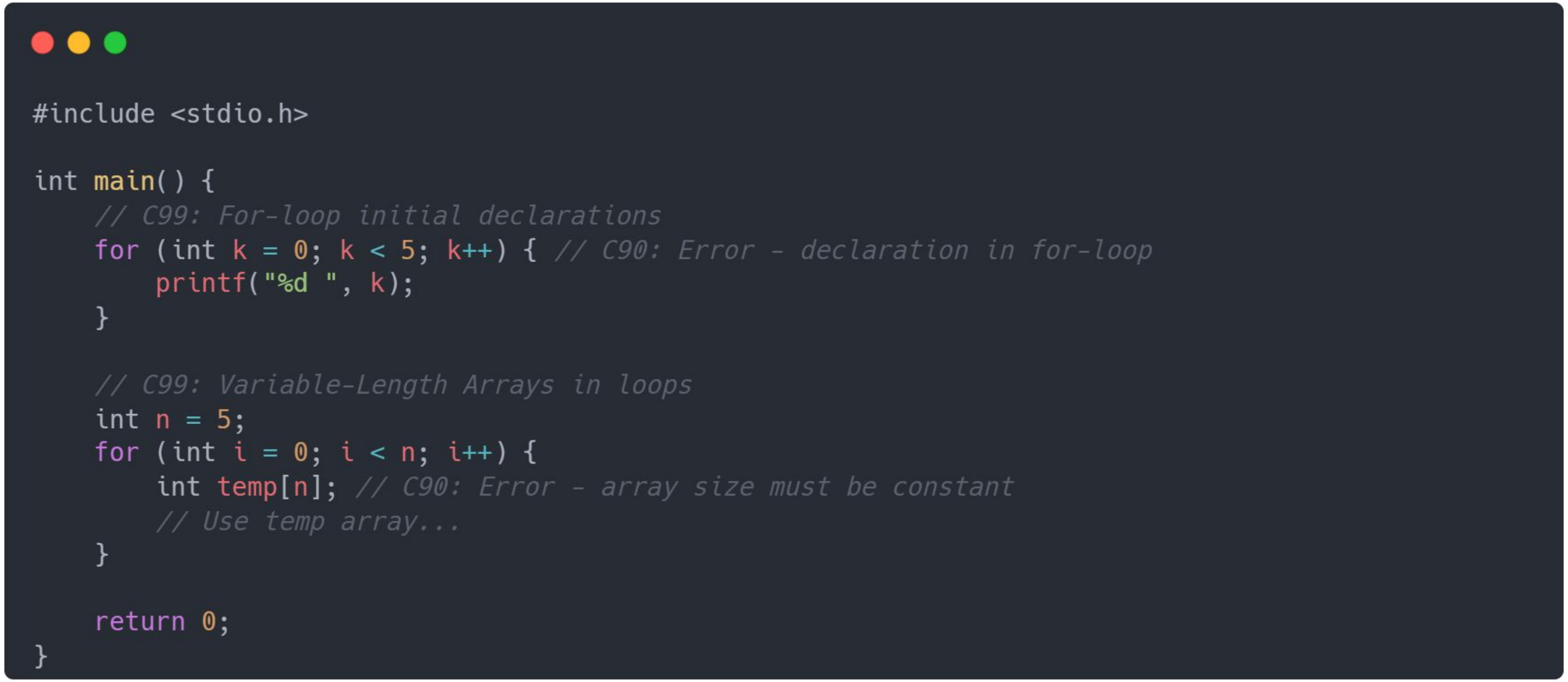


```c
int main() {
int nums[MAX];
int size=0;  // Bad: No space around = operator

printf("Enter up to %d numbers (enter -1 to stop):\n",MAX);
while(size<MAX) {
int num;scanf("%d",&num);  // Bad: Multiple statements on one line
if(num==-1)break;  // Bad: No spaces around operators, no braces for single-line if body
nums[size++]=num;}  // Bad: Closing brace on same line as code

printf("The array contains:\n");
print_arr(nums,size);  // Bad: No space after comma

int total=sumArr(nums,size);  // Bad: No spaces around = operator
printf("The sum of all elements is: %d\n",total);

return 0;
} // Bad: No space after last function
```

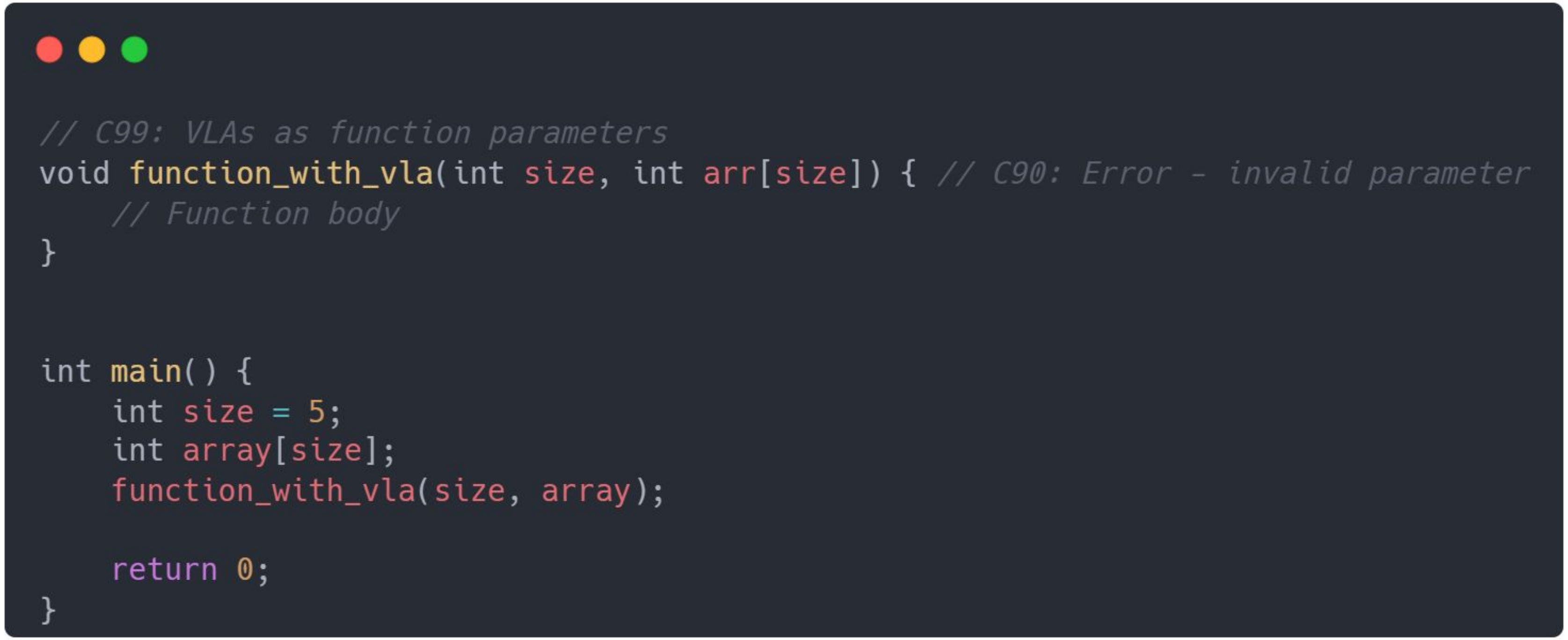
C90 vs C99



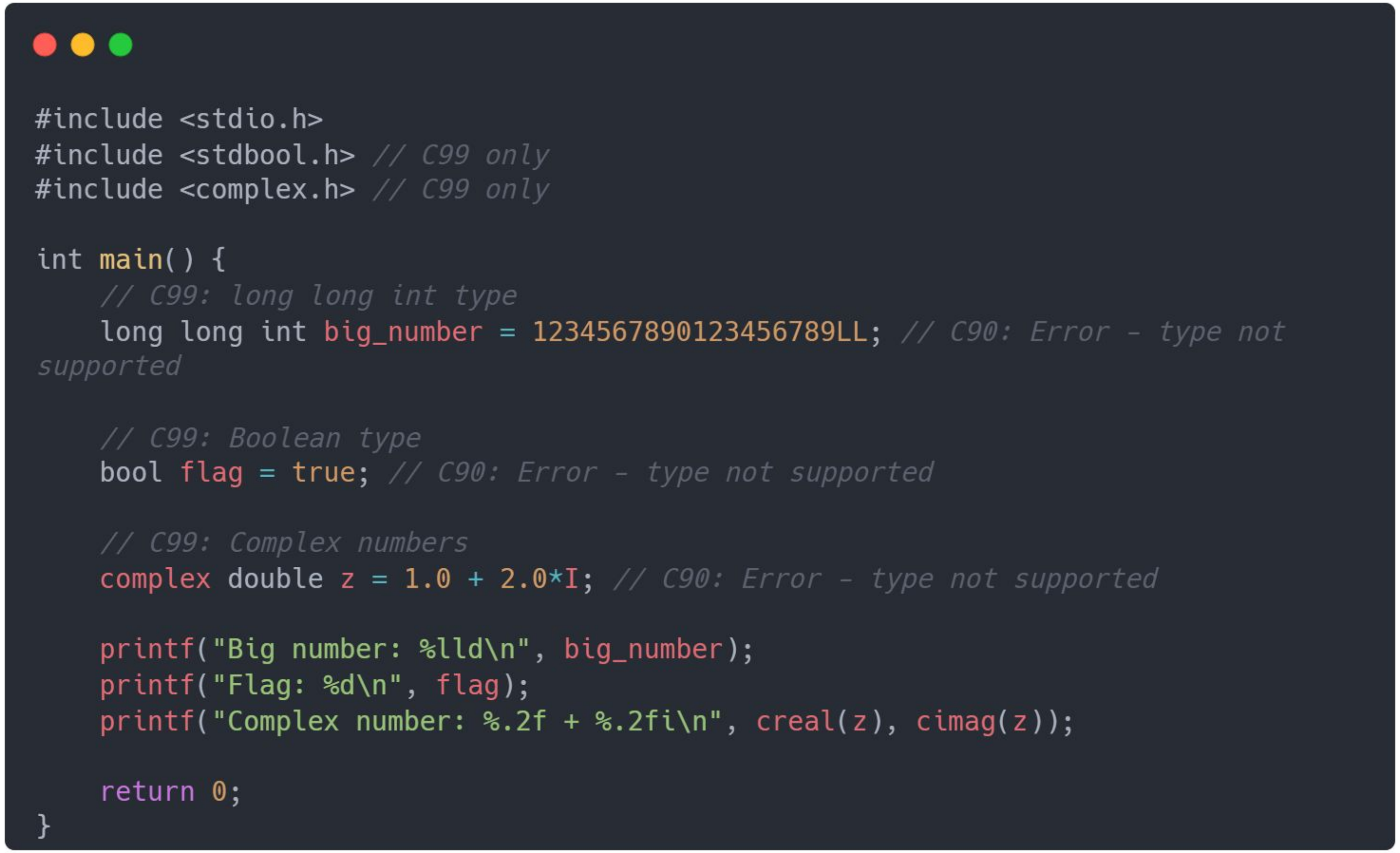
```c
#include <stdio.h>

int main() {
    // C99: Variable declarations mid-block
    int i = 0;
    printf("i is %d\n", i);

    int j = 1; // C90: Error - declaration not at start of block

    // C99: Variable-Length Arrays (VLAs)
    int n = 5;
    int vla[n]; // C90: Error - array size must be constant

    // C99: Designated initializers
    int arr[5] = {[2] = 5, [4] = 10}; // C90: Error - invalid syntax

    // C99: Variable-length multidimensional arrays
    int rows = 3, cols = 4;
    int matrix[rows][cols]; // C90: Error - array size must be constant

    return 0;
}
```



```c
#include <stdio.h>

int main() {
    // C99: For-loop initial declarations
    for (int k = 0; k < 5; k++) { // C90: Error - declaration in for-loop
        printf("%d ", k);
    }

    // C99: Variable-Length Arrays in loops
    int n = 5;
    for (int i = 0; i < n; i++) {
        int temp[n]; // C90: Error - array size must be constant
        // Use temp array...
    }

    return 0;
}
```



```c
// C99: VLAs as function parameters
void function_with_vla(int size, int arr[size]) { // C90: Error - invalid parameter
    // Function body
}


int main() {
    int size = 5;
    int array[size];
    function_with_vla(size, array);

    return 0;
}
```



```c
#include <stdio.h>
#include <stdbool.h> // C99 only
#include <complex.h> // C99 only

int main() {
    // C99: long long int type
    long long int big_number = 1234567890123456789LL; // C90: Error - type not
supported

    // C99: Boolean type
    bool flag = true; // C90: Error - type not supported

    // C99: Complex numbers
    complex double z = 1.0 + 2.0*I; // C90: Error - type not supported

    printf("Big number: %lld\n", big_number);
    printf("Flag: %d\n", flag);
    printf("Complex number: %.2f + %.2fi\n", creal(z), cimag(z));

    return 0;
}
```