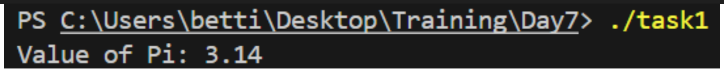
1.

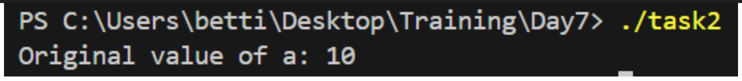
```c
/*Assignment 1: Constant Variable Declaration
Objective: Learn to declare and initialize constant variables.
Write a program that declares a constant integer variable for the value of Pi
(3.14) and prints it. Ensure that any attempt to modify this variable results in
a compile-time error. */

#include <stdio.h>
float const Pi = 3.14;
int main() {
    printf("Value of Pi: %.2f\n", Pi);
    // Pi = 3.15; // Attempt to modify the constant variable
    return 0;
}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task1
Value of Pi: 3.14
```

2.

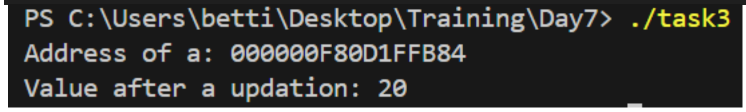
```c
/*Assignment 2: Using const with Pointers
Objective: Understand how to use const with pointers to prevent modification of
pointed values.
Create a program that uses a pointer to a constant integer. Attempt to modify the
value through the pointer and observe the compiler's response. */

#include <stdio.h>
int main()
{
    int a = 10;
    int const *pData = (int*) &a;

    printf("Original value of a: %d\n", *pData);

    //*pData = 0; // error: assignment of read-only location '*pData'

    // printf("Modified value of a through pointer: %d\n", a);

    return 0;
}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task2
Original value of a: 10
```

3.

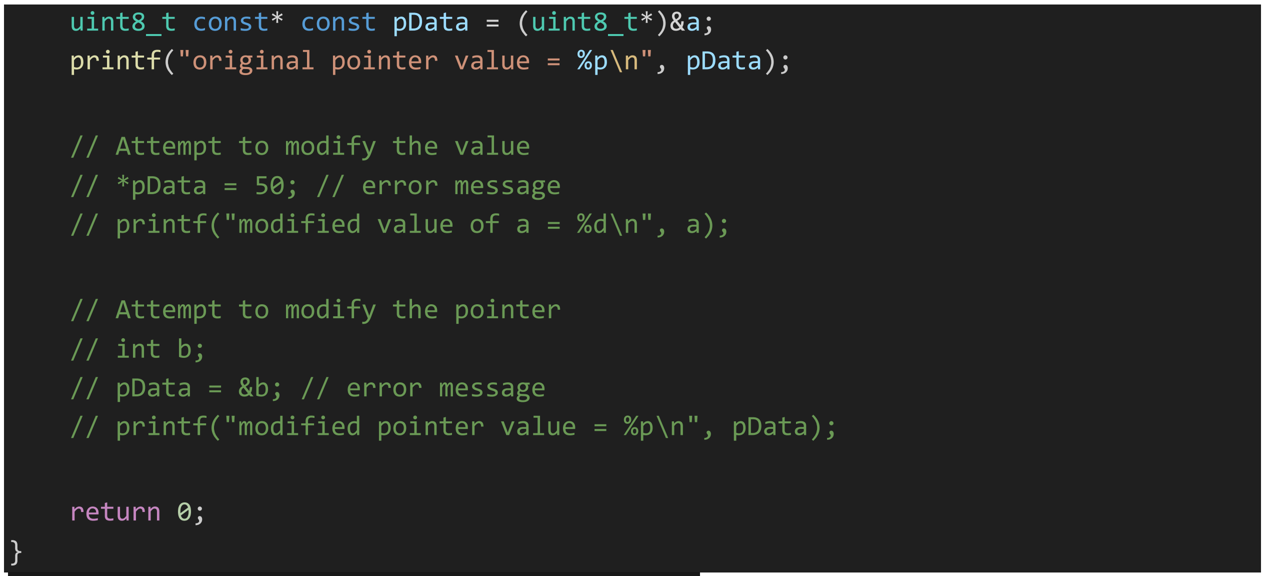
```c
/*Assignment 3: Constant Pointer
Objective: Learn about constant pointers and their usage.
Write a program that declares a constant pointer to an integer and demonstrates
that you cannot change the address stored in the pointer. */

#include <stdio.h>
#include <stdint.h>
int main() {
    int a = 10;
    uint8_t *const pData = (uint8_t*)&a;
    printf("Address of a: %p\n", pData);

    // int b;
    // pData = (int*)&b; //error: assignment of read-only variable 'pData'
    // printf("Address of b after changing the pointer: %d\n", pData);

    *pData = a + 10; // ponted value can be updated but not the address stored in
the pointer
    printf("Value after a updation: %d\n", a);

    return 0;
}
```



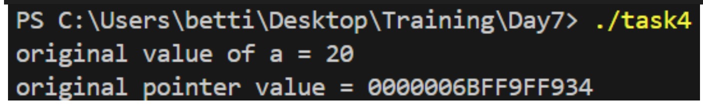
```
PS C:\Users\betti\Desktop\Training\Day7> ./task3
Address of a: 000000F80D1FFB84
Value after a updation: 20
```

4.

```c
/*Assignment 4: Constant Pointer to Constant Value
Objective: Combine both constant pointers and constant values.
Create a program that declares a constant pointer to a constant integer.
Demonstrate that neither the pointer nor the value it points to can be changed.
*/

#include <stdio.h>
#include <stdint.h>
int main(){
    int a = 20;
    printf("original value of a = %d\n", a);
```

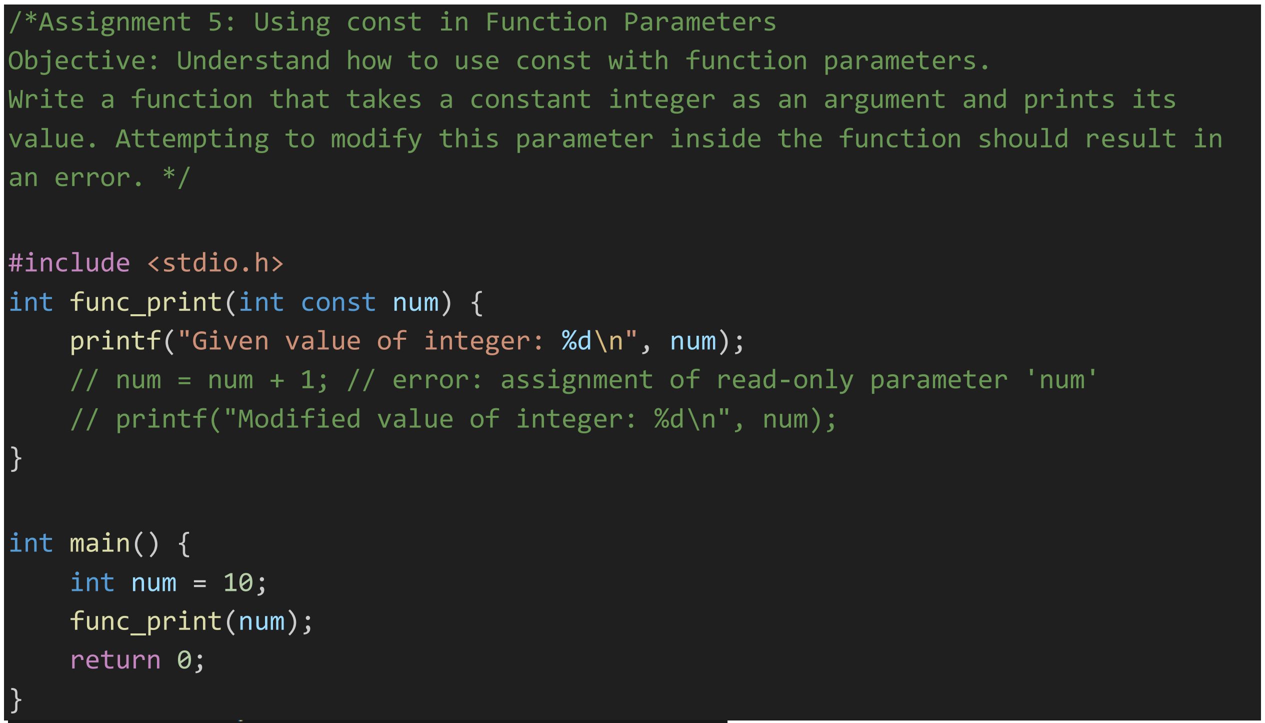


```c
    uint8_t const* const pData = (uint8_t*)&a;
    printf("original pointer value = %p\n", pData);

    // Attempt to modify the value
    // *pData = 50; // error message
    // printf("modified value of a = %d\n", a);

    // Attempt to modify the pointer
    // int b;
    // pData = &b; // error message
    // printf("modified pointer value = %p\n", pData);

    return 0;
}
```

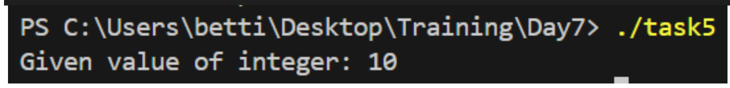


```
PS C:\Users\betti\Desktop\Training\Day7> ./task4
original value of a = 20
original pointer value = 0000006BFF9FF934
```

5.



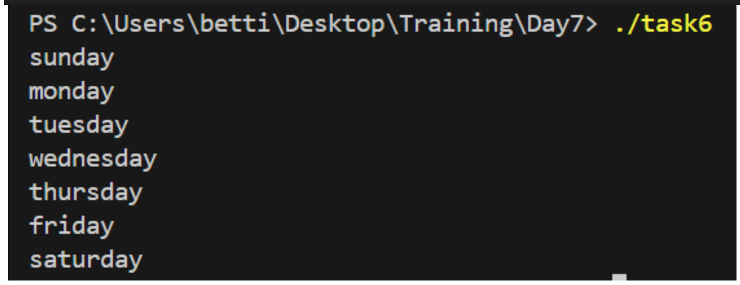
```c
/*Assignment 5: Using const in Function Parameters
Objective: Understand how to use const with function parameters.
Write a function that takes a constant integer as an argument and prints its
value. Attempting to modify this parameter inside the function should result in
an error. */

#include <stdio.h>
int func_print(int const num) {
    printf("Given value of integer: %d\n", num);
    // num = num + 1; // error: assignment of read-only parameter 'num'
    // printf("Modified value of integer: %d\n", num);
}

int main() {
    int num = 10;
    func_print(num);
    return 0;
}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task5
Given value of integer: 10
```

6.

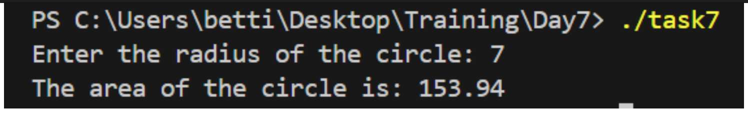
```c
/*Assignment 6: Array of Constants
Objective: Learn how to declare and use arrays with const.
Create an array of constants representing days of the week. Print each day using
a loop, ensuring that no modifications can be made to the array elements. */

#include <stdio.h>

int main()
{
    char const *days_of_week[] ={"sunday", "monday", "tuesday", "wednesday",
"thursday", "friday", "saturday"};

    for (int i = 0; i < 7; i++) {
        printf("%s\n", days_of_week[i]);
        // *days_of_week[i] = "hello"; // error: assignment of read-only location
'*days_of_week[i]'
    }

    return 0;
}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task6
sunday
monday
tuesday
wednesday
thursday
friday
saturday
```

7.

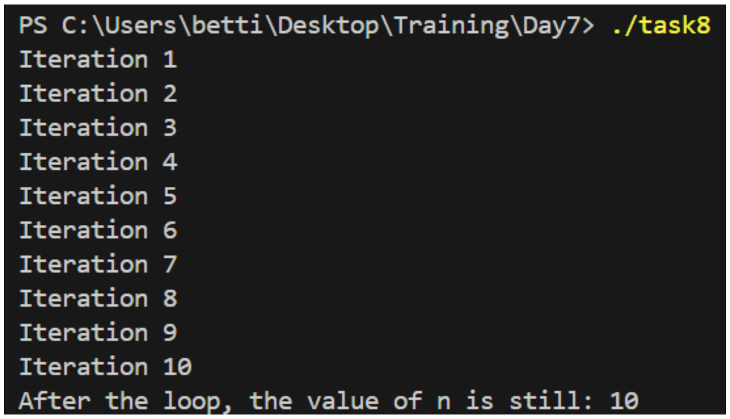
```c
/*Assignment 7: Constant Expressions
Objective: Understand how constants can be used in expressions.
Write a program that uses constants in calculations, such as calculating the area
of a circle using const.*/

#include <stdio.h>
int main()
{
    float radius;
    printf("Enter the radius of the circle: ");
    scanf("%f", &radius);

    float const area = 3.14159 * radius * radius;
    printf("The area of the circle is: %.2f\n", area);

    // area = area * 4; // error: assignment of read-only variable 'area'
    // printf("The area of the circle if radius is doubled: %.2f\n", area);

    return 0;
}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task7
Enter the radius of the circle: 7
The area of the circle is: 153.94
```

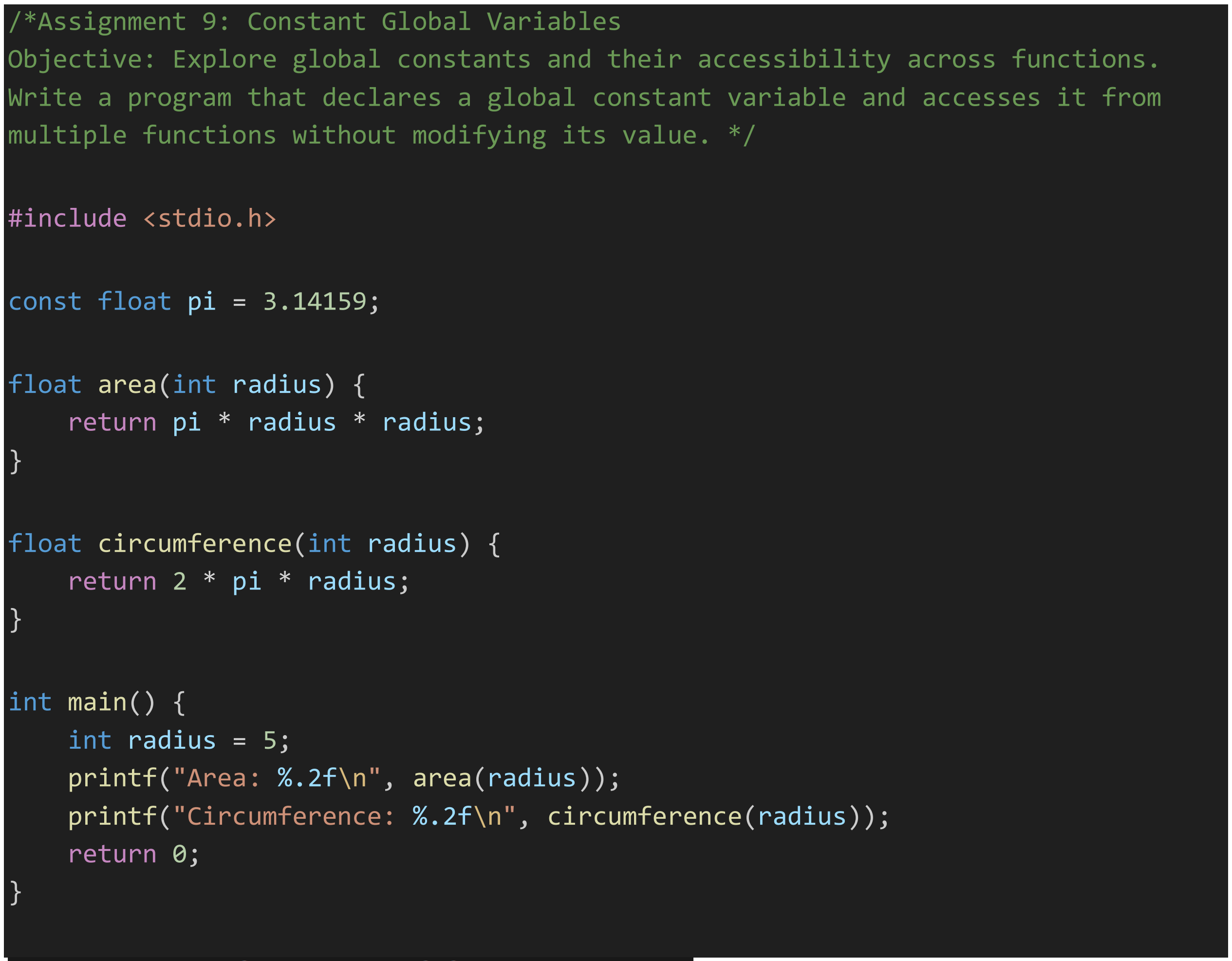
8.

```c
/*Assignment 8: Constant Variables in Loops
Objective: Learn how constants can be used within loops for fixed iterations.
Create a program that uses a constant variable to define the number of iterations
in a loop, ensuring it cannot be modified during execution. */

#include <stdio.h>

int main() {

    int const n = 10;

    for (int i = 0; i < n; i++) {
        printf("Iteration %d\n", i + 1);
        // n = i + 1; // error: assignment of read-only variable 'n'
    }

    printf("After the loop, the value of n is still: %d\n", n);

    return 0;
}
```

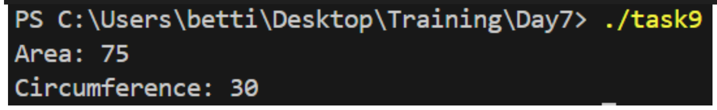


```
PS C:\Users\betti\Desktop\Training\Day7> ./task8
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8
Iteration 9
Iteration 10
After the loop, the value of n is still: 10
```

9.

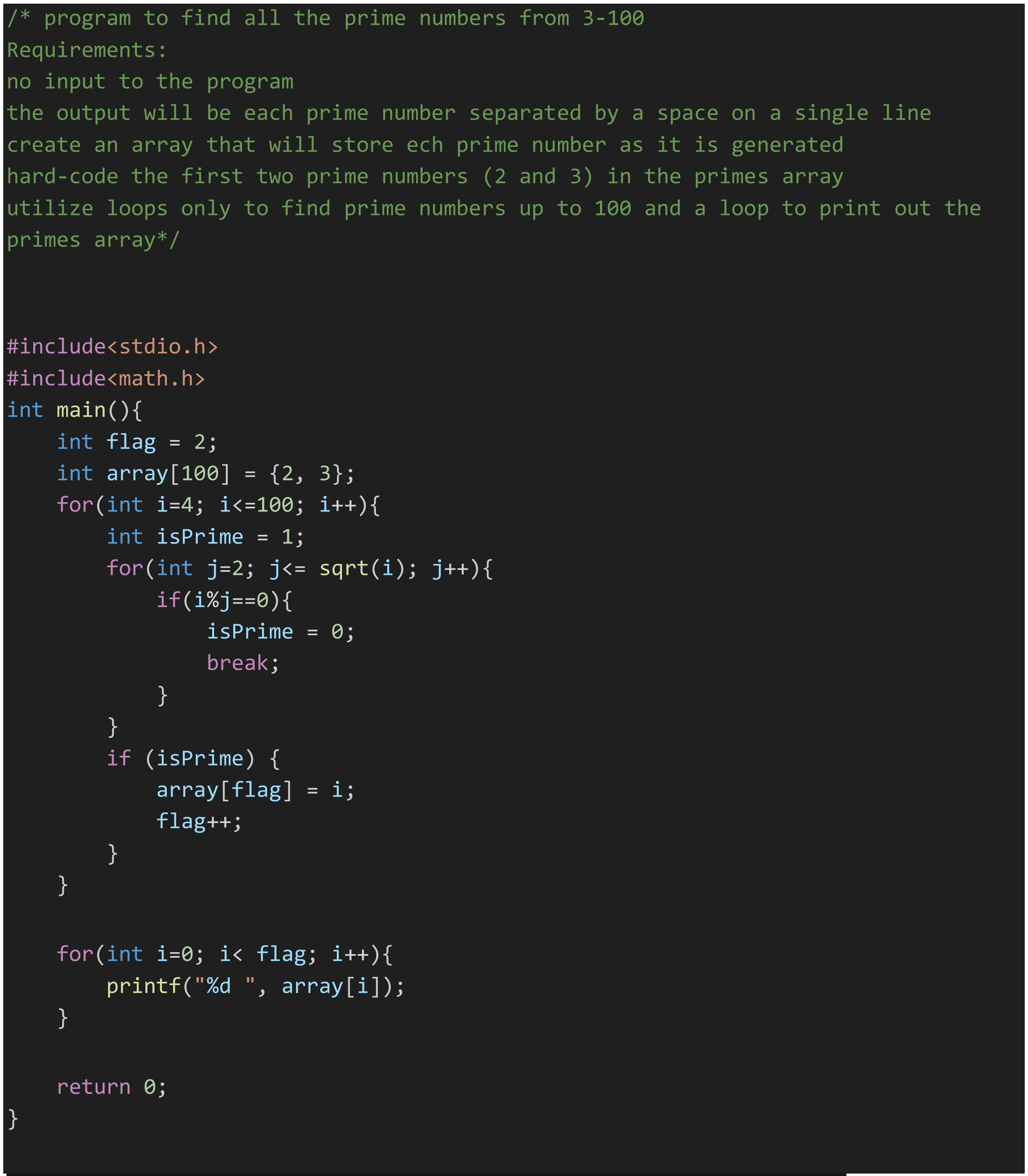


```c
/*Assignment 9: Constant Global Variables
Objective: Explore global constants and their accessibility across functions.
Write a program that declares a global constant variable and accesses it from
multiple functions without modifying its value. */

#include <stdio.h>

const float pi = 3.14159;

float area(int radius) {
    return pi * radius * radius;
}

float circumference(int radius) {
    return 2 * pi * radius;
}

int main() {
    int radius = 5;
    printf("Area: %.2f\n", area(radius));
    printf("Circumference: %.2f\n", circumference(radius));
    return 0;
}
```

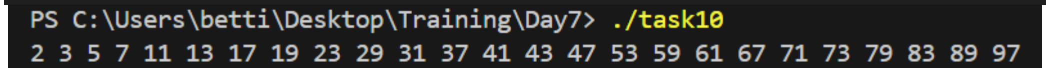


```
PS C:\Users\betti\Desktop\Training\Day7> ./task9
Area: 75
Circumference: 30
```

10.



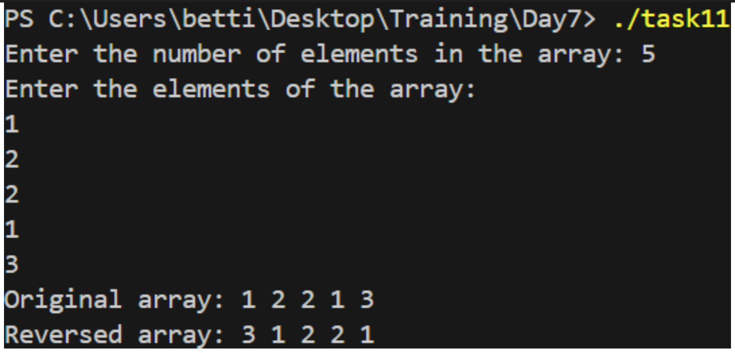
```c
/* program to find all the prime numbers from 3-100
Requirements:
no input to the program
the output will be each prime number separated by a space on a single line
create an array that will store ech prime number as it is generated
hard-code the first two prime numbers (2 and 3) in the primes array
utilize loops only to find prime numbers up to 100 and a loop to print out the
primes array*/


#include<stdio.h>
#include<math.h>
int main(){
    int flag = 2;
    int array[100] = {2, 3};
    for(int i=4; i<=100; i++){
        int isPrime = 1;
        for(int j=2; j<= sqrt(i); j++){
            if(i%j==0){
                isPrime = 0;
                break;
            }
        }
        if (isPrime) {
            array[flag] = i;
            flag++;
        }
    }

    for(int i=0; i< flag; i++){
        printf("%d ", array[i]);
    }

    return 0;
}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task10
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

11.

```c
// Create a program that reverses the elements of an array. Prompt the user to
enter values and print both the original and reversed arrays.

#include <stdio.h>
int main(){
    int n;
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter the elements of the array:\n");
    for(int i = 0; i < n; i++){
        scanf("%d", &arr[i]);
    }

    printf("Original array: ");
    for(int i = 0; i < n; i++){
        printf("%d ", arr[i]);
    }

    printf("\nReversed array: ");
    for(int i = n-1; i >= 0; i--){
        printf("%d ", arr[i]);
    }

    return 0;
}
```

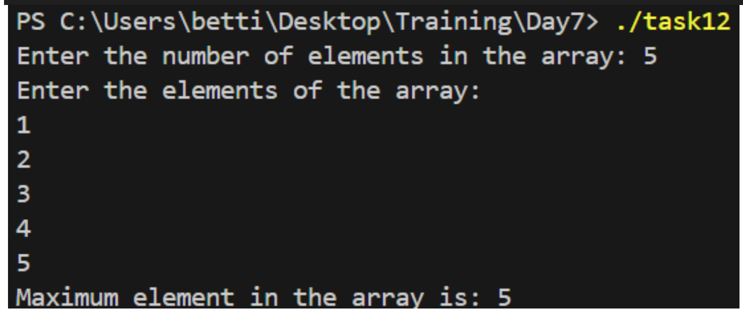


```
PS C:\Users\betti\Desktop\Training\Day7> ./task11
Enter the number of elements in the array: 5
Enter the elements of the array:
1
2
2
1
3
Original array: 1 2 2 1 3
Reversed array: 3 1 2 2 1
```

12.

```c
//Write a program that to find the maximum element in an array of integers. The
program should prompt the user for input and display the maximum value.
```

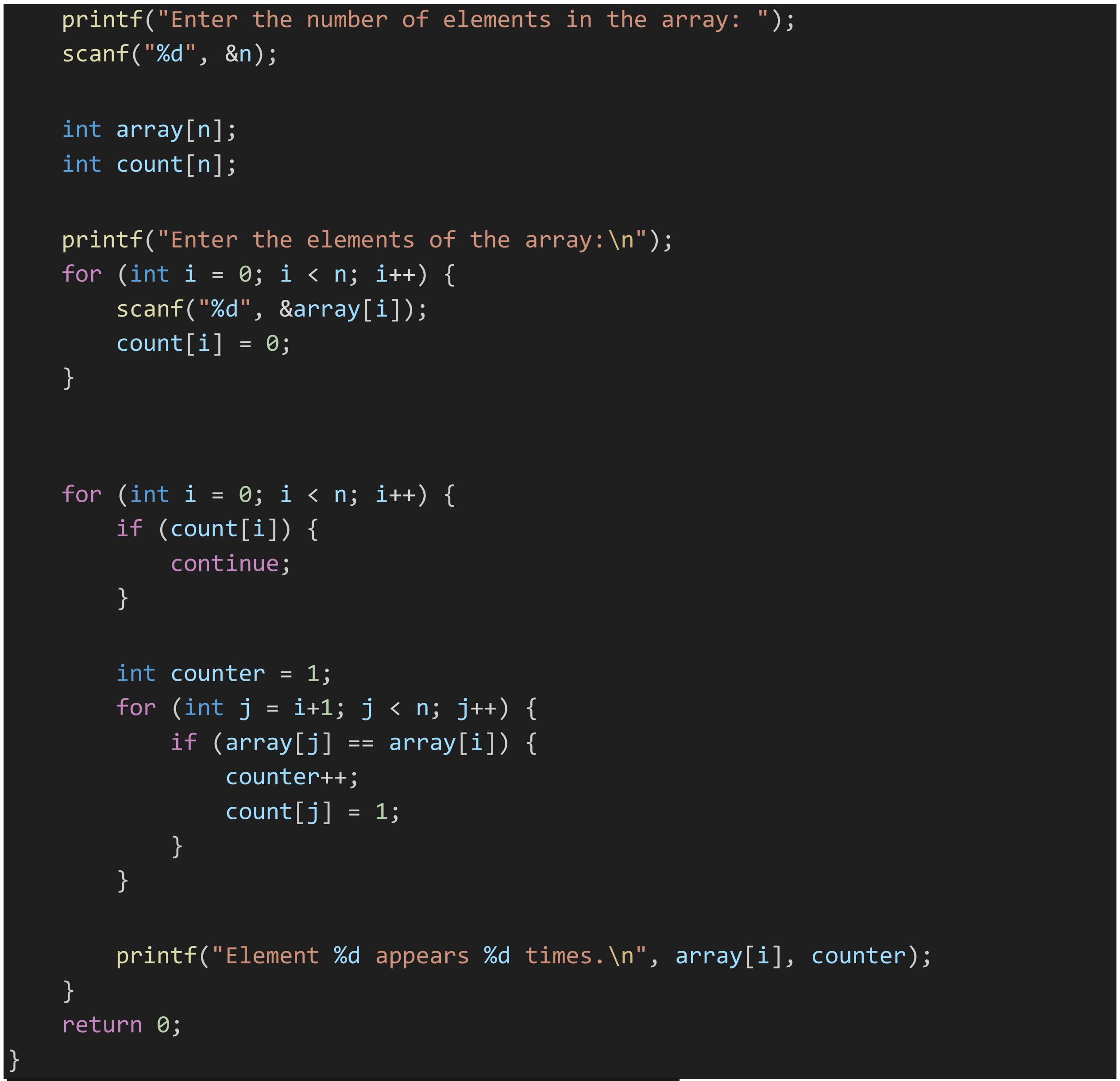
```c
#include<stdio.h>
int main() {
    int n, max;
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);

    int array[n];
    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &array[i]);
    }

    max = array[0];
    for (int i = 1; i < n; i++) {
        if( array[i] > max )
        {
            max = array[i];
        }
    }

    printf("Maximum element in the array is: %d\n", max);

    return 0;
}
```



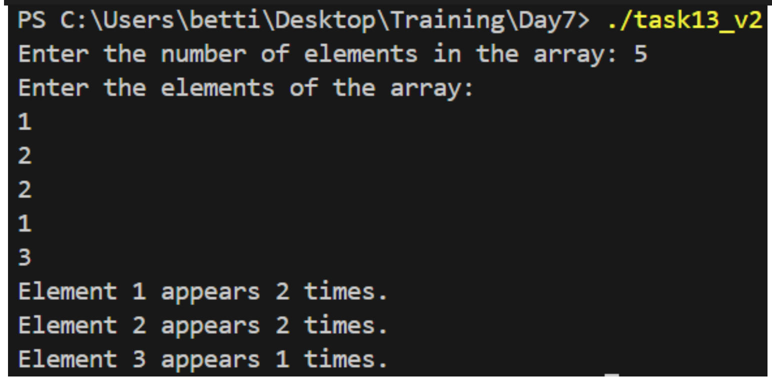
```
PS C:\Users\betti\Desktop\Training\Day7> ./task12
Enter the number of elements in the array: 5
Enter the elements of the array:
1
2
3
4
5
Maximum element in the array is: 5
```

13.

```c
//Write a program that counts and displays how many times a specific integer
appears in an array entered by the user.

#include <stdio.h>

int main() {
    int n;
```



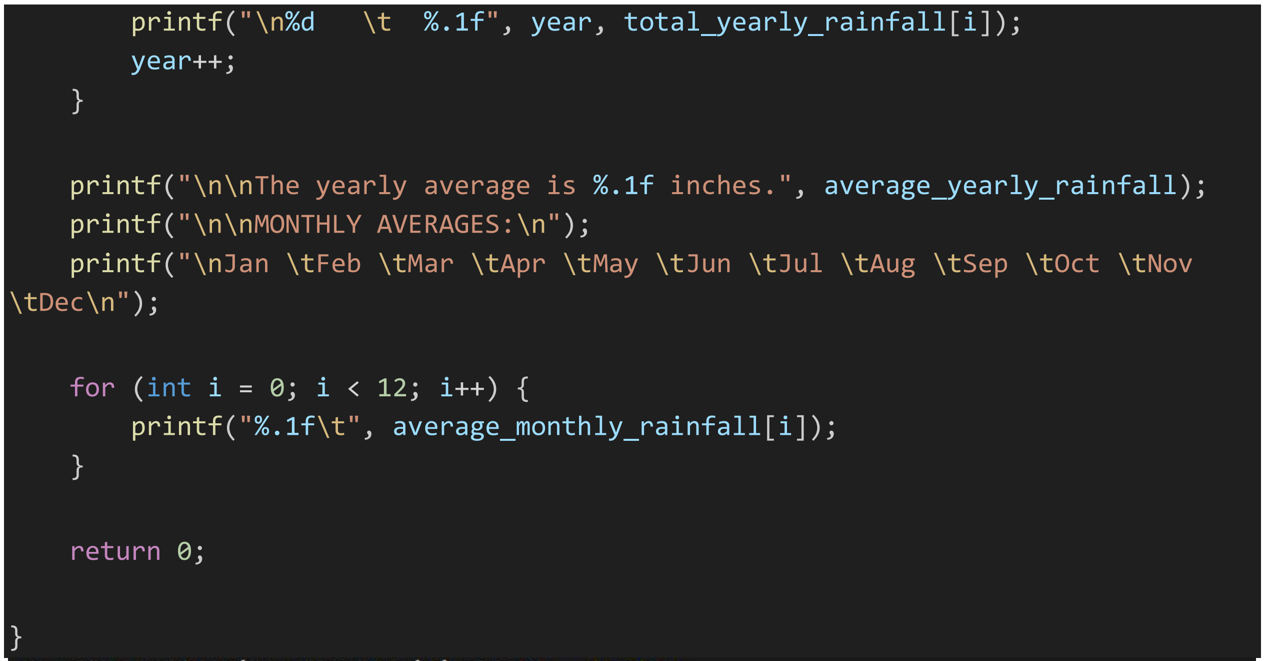
```c
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);

    int array[n];
    int count[n];

    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &array[i]);
        count[i] = 0;
    }



    for (int i = 0; i < n; i++) {
        if (count[i]) {
            continue;
        }

        int counter = 1;
        for (int j = i+1; j < n; j++) {
            if (array[j] == array[i]) {
                counter++;
                count[j] = 1;
            }
        }

        printf("Element %d appears %d times.\n", array[i], counter);
    }
    return 0;
}
```



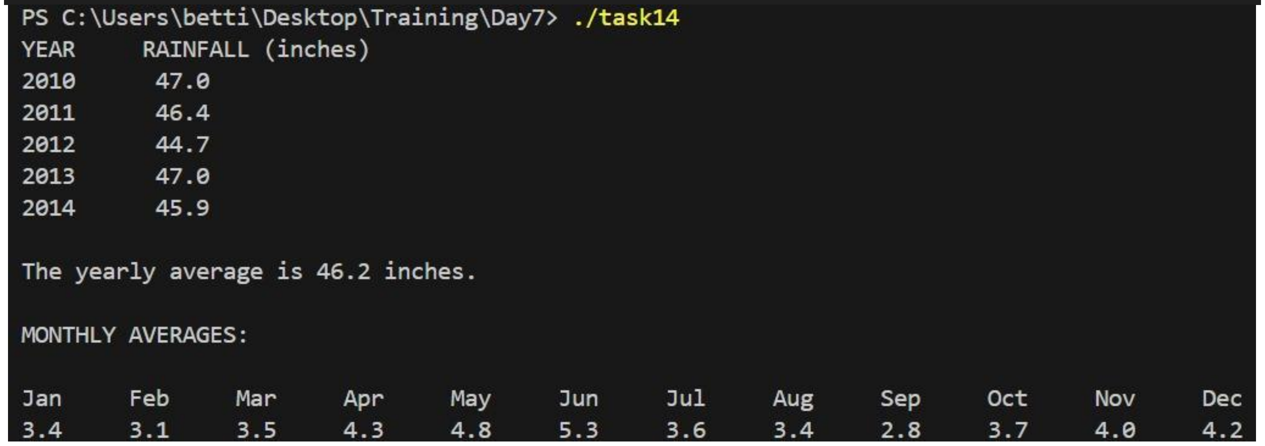
```
PS C:\Users\betti\Desktop\Training\Day7> ./task13_v2
Enter the number of elements in the array: 5
Enter the elements of the array:
1
2
2
1
3
Element 1 appears 2 times.
Element 2 appears 2 times.
Element 3 appears 1 times.
```

14.

```c
/*Create a program that uses two-dimensional array in weather program.
This program will find the total rainfall for each year, the average yearly
rainfall, and the average rainfall for each month.
Input will be a 2D array with hard-corded values for rainfall amounts for the
past 5 years.
The array should have 5 rows and 12 columns.
Rainfall amounts can be floating point numbers. */

#include<stdio.h>
int main() {
    float total_yearly_rainfall[5] = {0.0},total_rainfall = 0.0,
average_yearly_rainfall = 0.0, average_monthly_rainfall[12] = {0.0};
    float rainfall[5][12] = {
        {3.36, 3.08, 3.63, 4.33, 4.86, 5.52, 3.77, 3.49, 2.97, 3.87, 3.95, 4.16},
        {3.55, 3.17, 3.53, 4.54, 4.76, 5.14, 3.56, 3.35, 2.77, 3.74, 4.04, 4.28},
        {3.10, 3.23, 3.48, 4.15, 4.70, 5.31, 3.35, 3.18, 2.58, 3.58, 3.91, 4.11},
        {3.50, 3.02, 3.56, 4.42, 4.95, 5.45, 3.64, 3.45, 2.85, 3.85, 4.00, 4.33},
        {3.28, 3.20, 3.46, 4.29, 4.80, 5.30, 3.67, 3.34, 2.93, 3.70, 3.89, 4.08}
    };



    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 12; j++) {
            total_yearly_rainfall[i] += rainfall[i][j];
        }
    }

    for (int i = 0; i < 5; i++) {
        total_rainfall += total_yearly_rainfall[i];
    }

    average_yearly_rainfall = total_rainfall / 5;

    for (int i = 0; i < 12; i++) {
        for (int j = 0; j < 5; j++) {
            average_monthly_rainfall[i] += rainfall[j][i];
        }
        average_monthly_rainfall[i] /= 5;
    }

    int year = 2010;
    printf("YEAR \t RAINFALL (inches)");
    for (int i = 0; i < 5; i++) {
```



```c
        printf("\n%d   \t  %.1f", year, total_yearly_rainfall[i]);
        year++;
    }

    printf("\n\nThe yearly average is %.1f inches.", average_yearly_rainfall);
    printf("\n\nMONTHLY AVERAGES:\n");
    printf("\nJan \tFeb \tMar \tApr \tMay \tJun \tJul \tAug \tSep \tOct \tNov 
\tDec\n");

    for (int i = 0; i < 12; i++) {
        printf("%.1f\t", average_monthly_rainfall[i]);
    }

    return 0;

}
```



```
PS C:\Users\betti\Desktop\Training\Day7> ./task14
YEAR     RAINFALL (inches)
2010      47.0
2011      46.4
2012      44.7
2013      47.0
2014      45.9

The yearly average is 46.2 inches.

MONTHLY AVERAGES:

Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct     Nov     Dec
3.4     3.1     3.5     4.3     4.8     5.3     3.6     3.4     2.8     3.7     4.0     4.2
```