Day7_assignment_Abhirami

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 a=3.14;

int main()

{

    printf("a=%f\n",a);

    a=45.7;

    printf("a=%f\n",a);

    return 0;

}
```

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.

```c
#include <stdio.h>

int const a=50;

int main()

{

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

    int *p;

    p=&a;

    *p=80;

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

    return 0;

}
```

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.

```c
#include <stdio.h>

int main() {

    int a = 10;

    int b = 20;

  int *const ptr = &a;

printf("Initial value: %d\n", *ptr);

 *ptr = 15;

 printf("Modified value: %d\n", *ptr);

// ptr = &b;

return 0;
```

```
}
```

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>

const int a = 10;

int main() {

    const int *const ptr = &a;

printf("Value of a: %d\n", *ptr);

// Attempting to modify the value

    //*ptr = 15;

 // Attempting to change the pointer

  // int b = 20;

  // ptr = &b;

 return 0;

}
```

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>

const int num = 10;

void printValue(const int a) {
```

```
printf("Value: %d\n", a);
    //modify
    // a = 20;
}

int main() {
    printValue(num);
return 0;
}
```

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() {
    const char *daysOfWeek[] = {
        "Sunday", "Monday", "Tuesday", "Wednesday",
        "Thursday", "Friday", "Saturday"
    };
 for (int i = 0; i < 7; i++) {
        printf("%s\n", daysOfWeek[i]);
 }
 return 0;
}
```

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>

const double PI = 3.14159;

const double RADIUS = 5.0;

int main() {

    double area = PI * RADIUS * RADIUS;

 printf("Area of the circle with radius %.2f is: %.2f\n", RADIUS, area);

return 0;

}
```

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>

const int count = 5;

int main() {

for (int i = 1; i <= count; i++) {

        printf("Iteration %d\n", i);

    }

  return 0;

}
```

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 int a = 100;

void printA() {

    printf("The value of a is: %d\n", a);

}

void doubleA() {

    printf("Double the value of a is: %d\n", a * 2);

}

int main() {

    printA();

    doubleA();

return 0;

}
```

//Requirements

• In this challenge, you are going to create a program that will find all the prime numbers from 3-100

• there will be no input to the program

• The output will be each prime number separated by a space on a single line

• You will need to create an array that will store each prime number as it is generated

• You can hard-code the first two prime numbers (2 and 3) in the primes array

• You should utilize loops to only find prime numbers up to 100 and a loop to print out the primes array

```
#include<stdio.h>
int main() {
    int primes[100];
    int count = 2;
 primes[0] = 2;
    primes[1] = 3;
for (int num = 5; num <= 100; num += 2) {
        int isPrime = 1;
 for (int i = 1; i < count; i++) {
            if (num % primes[i] == 0) {
                isPrime = 0;
                break;
            }
        }

        if (isPrime) {
            primes[count] = num;
```

```
            count++;
        }
    }


    for (int i = 0; i < count; i++) {
        printf("%d ", primes[i]);
    }
 return 0;
}
```

1.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 size of the array: ");
    scanf("%d", &n);
int arr[n];
    printf("Enter the elements in the array:\n");
    for (int i = 0; i < n; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }
 // Reverse the array
    int arr2[n];
    for (int i = 0; i < n; i++) {
```

```
        arr2[i] = arr[n - 1 - i];
    }

    // Print reversed array
    printf("The reversed array elements are:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr2[i]);
    }
 return 0;
}
```

2. 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.

```
#include <stdio.h>
int main() {
    int n;
    printf("Enter the size of the array: ");
    scanf("%d", &n);
  int arr[n];
    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }
```

```
int max = arr[0];
    for (int i = 1; i < n; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
    }
printf("The maximum element in the array is: %d\n", max);
return 0;
}
```

3. 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, searchNum, count = 0;
printf("Enter the size of the array: ");
    scanf("%d", &n);
int arr[n];
 printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }
 printf("Enter the number to search for: ");
    scanf("%d", &searchNum);
```

```
    for (int i = 0; i < n; i++) {
        if (arr[i] == searchNum) {
            count++;
        }
    }
printf("The number %d appears %d time(s) in the array.\n", searchNum, count);
  return 0;
}
```

Requirements

• In this challenge, you are to create a C program that uses a two-dimensional array in a weather program.

array is %0

• 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-coded 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

Answer:

```
#include<stdio.h>
 #define YEARS 5
#define MONTHS 12
 int main() {
float rainfall[YEARS][MONTHS];
```

```
 float yearlyTotals[YEARS] = {0};
 float totalRainfall = 0;
 printf("Enter the rainfall data for each month (in inches):\n");
 for (int year = 0; year < YEARS; year++) {
 printf("Year 201%d:\n", year);
for (int month = 0; month < MONTHS; month++) {
printf(" Month %d: ", month + 1);
scanf("%f", &rainfall[year][month]);
 yearlyTotals[year] += rainfall[year][month];
 }
totalRainfall += yearlyTotals[year];
 }
 printf("\nYEAR RAINFALL (inches)\n");
 for (int year = 0; year < YEARS; year++) {
printf("201%d %.1f\n", year, yearlyTotals[year]);
 }
 float yearlyAverage = totalRainfall / YEARS;
printf("\nThe yearly average is %.1f inches.\n", yearlyAverage);
printf("\nMONTHLY AVERAGES:\n");
const char *months[MONTHS] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep",
"Oct", "Nov", "Dec"};
 for (int month = 0; month < MONTHS; month++) {
float monthlyTotal = 0;
for (int year = 0; year < YEARS; year++) {
monthlyTotal += rainfall[year][month];
```

```
}
printf("%s %.1f\n", months[month], monthlyTotal / YEARS);
 }
 return 0;
 }
```