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("pi=%f\n",pi);
   //pi=4.5;
   printf("pi=%f\n",pi);
   return 0;
}
```

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 const a=10;
int main()
{
   printf("a=%d\n",a);
   int *p;
   p=&a;
   //*p=20;
   printf("a=%d\n",a);
   return 0;
}
```

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>


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;
}
```

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);
   //*ptr = 20;
```

```
  //int b = 20;
  //ptr = &b;
  return 0;
}
```

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 func(const int a)
{
  printf("Value: %d\n", a);
  //a = 20;
}

int main()
{
  func(num);
  printf("new val: %d\n",num);
  return 0;
}
```

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 * const days[] = {
      "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};


   //days[0]="joel";


   for (int i = 0; i < 7; i++) {
      printf("%s\n", days[i]);
   }
   return 0;
}
```

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 float Pi = 3.14;
int main() {
   int rad=20;
   float area = Pi * rad * rad;
   printf("Area of the circle is: %f\n",area);


   return 0;
}
```

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 counter = 10;
int main() {
  for (int i = 1; i <= counter; i++)
  {
    printf("counter %d\n", i);
  }
  return 0;
}
```

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


void B()
{
  printf("Double the value of a is: %d\n", a * 2);
}
```

```
int main() {
  A();
  B();
 return 0;
}
```

10-)Initializing Arrays

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 checkprime(int n);
int main()
{
  int arr[100];
  int x=0;
  for(int i=3;i<100;i++)
  {
    int p=checkprime(i);
    if(p==1)
    {
      arr[x]=i;
      x=x+1;
    }
    }
    printf("2 ");
    for(int j=0;j<x;j++)
    {
        printf("%d ",arr[j]);
    }
    return 0;
}

int checkprime(int n)
    {
        int a=0;
        for(int i=2;i<n;i++)
        {
            if(n%i==0)
            {
                a=1;
            }
        }
        if(a==0)
        {
            return 1;
        }
        else
        {
            return 0;
        }
```

```
  }
```

12- 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 array: ");
  scanf("%d",&n);
  int arr[n];
  int rev[n];
  printf("Enter the elements\n");
  for(int i=0;i<n;i++)
  {
    scanf("%d",&arr[i]);
  }
  int a=0;
  for(int j=n-1;j>=0;j--)
  {
    rev[a]=arr[j];
    a=a+1;
  }
  printf("\n");
  for(int i=0;i<n;i++)
  {
    printf("%d ",rev[i]);
  }
  return 0;
}
```

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

    int max=arr[0];
    for(int j=1;j<n;j++)
    {
        if(arr[j]>max)
        {
            max=arr[j];
        }
    }
```

```
   printf("\n");
   printf("max element is %d",max);
   return 0;
}
```

14- 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;

   printf("Enter the number of elements in array: ");
   scanf("%d",&n);
   int arr[n];
   printf("Enter the elements\n");
   for(int i=0;i<n;i++)
   {
      scanf("%d",&arr[i]);
   }
   int num,cnt=0;
   printf("\n");
   printf("Enter the element to count: ");
   scanf("%d",&num);
   for(int i=0;i<n;i++)
   {
      if(arr[i]==num)
      {
        cnt=cnt+1;

      }

   }


   printf("the count is %d",cnt);

   return 0;

}
```

15-)Requirements

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

```
#include <stdio.h>

void main() {


   float rain_data[5][12] = {

      {4.5, 5.0, 6.2, 3.1, 4.8, 5.3, 6.1, 4.0, 3.7, 4.9, 5.3, 4.6},

      {4.2, 4.6, 6.0, 3.8, 4.9, 5.0, 5.8, 4.6, 3.9, 4.7, 4.9, 5.1},

      {3.8, 4.9, 5.7, 3.3, 4.6, 5.5, 6.0, 4.3, 3.6, 4.8, 5.1, 4.8},

      {4.0, 4.8, 6.3, 3.6, 4.7, 5.2, 5.9, 4.4, 3.8, 4.6, 5.0, 4.9},

      {4.3, 5.1, 6.1, 3.5, 4.5, 5.1, 5.7, 4.1, 3.9, 4.5, 5.2, 4.7}

   };


   float yearly_total_rain[5] = {0};
   float yearly_avg_rain[5] = {0};
   float monthly_avg_rain[12] = {0};

   char months[12][4] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun",
                         "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};


   for (int i = 0; i < 5; i++) {
      for (int j = 0; j < 12; j++) {
          yearly_total_rain[i] += rain_data[i][j];
          monthly_avg_rain[j] += rain_data[i][j];
      }
      yearly_avg_rain[i] = yearly_total_rain[i] / 12;
   }


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


   printf("Total rainfall for each year:\n");
   for (int i = 0; i < 5; i++) {
      printf("Year 202%d: %.2f\n", i, yearly_total_rain[i]);
   }

   printf("\nAverage rainfall for each year:\n");
   for (int i = 0; i < 5; i++) {
          printf("Year 202%d: %.2f\n", i, yearly_avg_rain[i]);
      }

      printf("\nAverage monthly rainfall over 5 years:\n");
      for (int j = 0; j < 12; j++) {
          printf("%s: %.2f\n", months[j], monthly_avg_rain[j]);
      }
   }
```