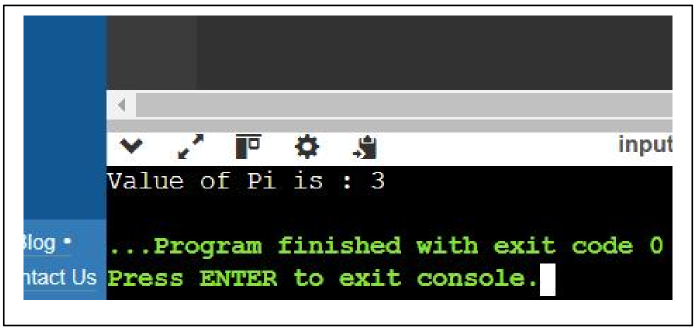
ASSIGNMENT

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.

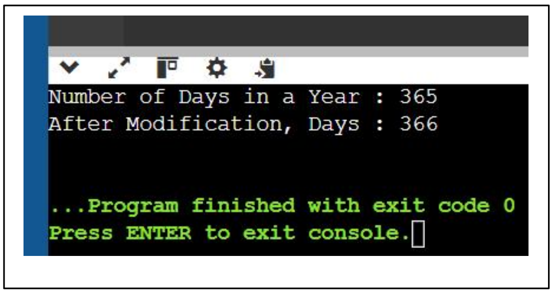
```
#include<stdio.h>

int const Pi=3.14;

int main(){

   // Pi=4;

   printf("Value of Pi is : %d",Pi);

   return 0;

}
```



```
Value of Pi is : 3

...Program finished with exit code 0
Press ENTER to exit console.
```

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.

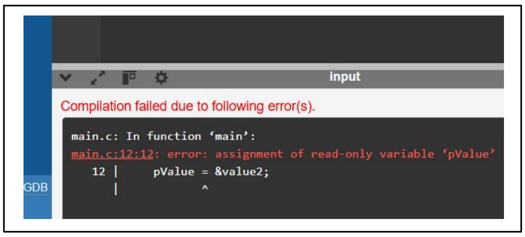
```
#include <stdio.h>

int main() {
   int days = 365;
   int const *pData = &days;


   printf("Number of Days in a Year : %d\n", *pData);

   days=366;
   printf("After Modification, Days : %d\n", *pData);

   return 0;
}
```



```
Number of Days in a Year : 365
After Modification, Days : 366


...Program finished with exit code 0
Press ENTER to exit console.
```

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>

int main() {
    int value1 = 10;
    int value2 = 20;

    int *const pValue = &value1;

    printf("Value in Pointer : %d\n",*pValue);
    pValue = &value2;
    printf("Modified Value of Pointer: %d\n", *pValue);

    return 0;
}
```

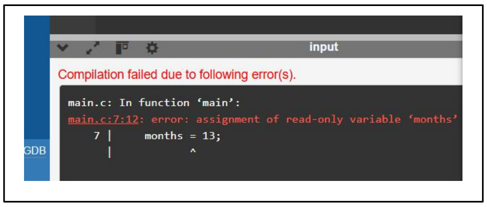


Compilation failed due to following error(s).

```
main.c: In function ‘main’:
main.c:12:12: error: assignment of read-only variable ‘pValue’
   12 |     pValue = &value2;
      |            ^
```

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>

int main() {
    const int months = 12;
    int const* const pData = &months;

    months = 13;
    printf("Number of Months in a Year: %d\n", *pData);

    return 0;
}
```

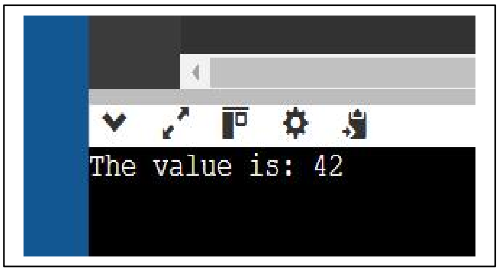


Compilation failed due to following error(s).

```
main.c: In function ‘main’:
main.c:7:12: error: assignment of read-only variable ‘months’
    7 |     months = 13;
      |            ^
```

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>

void printValue(const int value) {

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

    // value = 50;
```

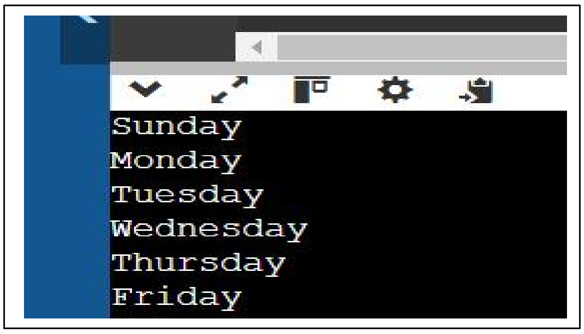


```
The value is: 42
```

```
}

int main() {

   int num = 42;

   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.

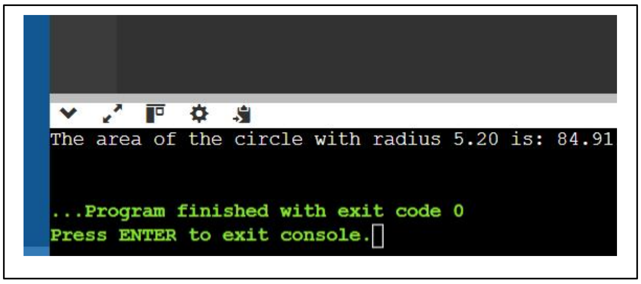
```
 #include <stdio.h>

int main() {

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

   for (int i = 0; i < 7; i++) {

      printf("%s\n", daysOfWeek[i]);

   }

   return 0;

}
```



```
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
```

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() {

   const float Pi = 3.14;

   const float radius = 5.2;
```

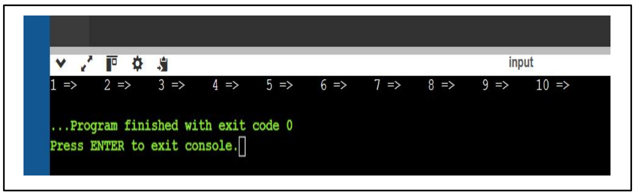


```
The area of the circle with radius 5.20 is: 84.91


...Program finished with exit code 0
Press ENTER to exit console.
```

```
    float area = Pi * radius * radius;

    printf("The 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.

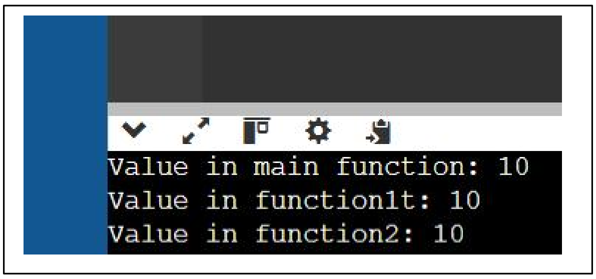
```
#include <stdio.h>
int main() {

    const int num = 10;

    for (int i = 1; i <= num; i++) {
        printf("%d =>\t", i);
    }

    return 0;
}
```



```
1 =>    2 =>    3 =>    4 =>    5 =>    6 =>    7 =>    8 =>    9 =>    10 =>

...Program finished with exit code 0
Press ENTER to exit console.
```

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 value = 10;
void function1() {
    printf("Value in function1t: %d\n", value);
}

void function2() {
    printf("Value in function2: %d\n", value);
}

int main() {
    printf("Value in main function: %d\n", value);

    function1();
    function2();
```

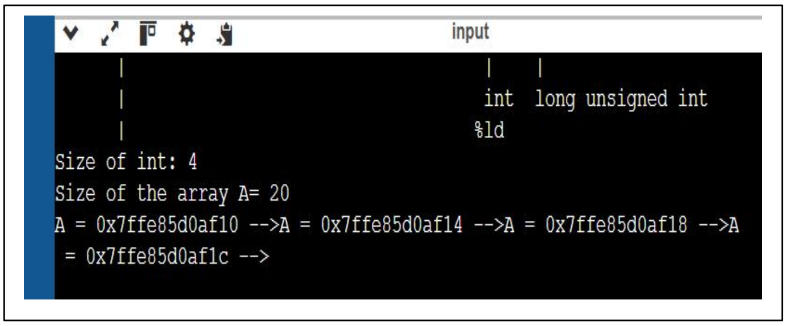


```
Value in main function: 10
Value in function1t: 10
Value in function2: 10
```

```
    return 0;
}
```

ARRAYS

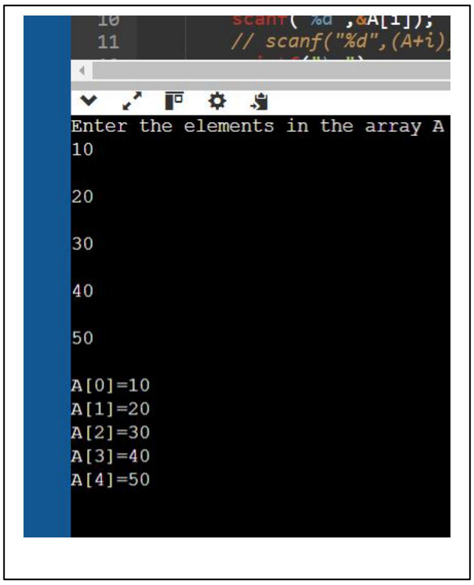
```
#include <stdio.h>

int main()
{
    int A[5];

    printf("Size of int: %d \n",sizeof(int));
    printf("Size of the array A= %d\n",sizeof(A));

    for(int i=0;i<4;i++){
        printf("A = %p -->",(A+i));      //(A+i)=Base address of Array + (index value * size
                                                                  //of the datatype)

    }
    return 0;
}
```



```
       |                                          |     |
       |                                          int  long unsigned int
       |                                         %ld
Size of int: 4
Size of the array A= 20
A = 0x7ffe85d0af10 -->A = 0x7ffe85d0af14 -->A = 0x7ffe85d0af18 -->A
 = 0x7ffe85d0af1c -->
```

Enter 5 elements into an array

```
#include<stdio.h>
int main(){

    int A[5];
    printf("Enter the elements in the array A \n");

    for(int i=0;i<5;i++){
        scanf("%d",&A[i]);
        // scanf("%d",(A+i));
        printf("\n");
    }
    for(int j=0;j<5;j++){
        printf("A[%d]=%d\n",j,A[j]);
    }
    return 0;
}
```



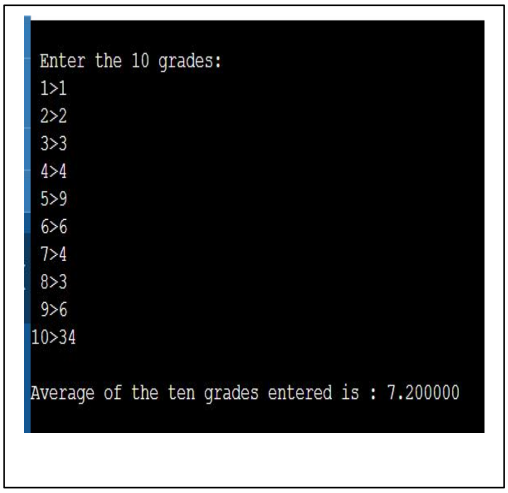
```
            scanf("%d",&A[i]);
            // scanf("%d",(A+i)
```

```
Enter the elements in the array A
10

20

30

40

50

A[0]=10
A[1]=20
A[2]=30
A[3]=40
A[4]=50
```

Average of 10 grades

```
#include <stdio.h>

int main()
{
    int grades[10];
```

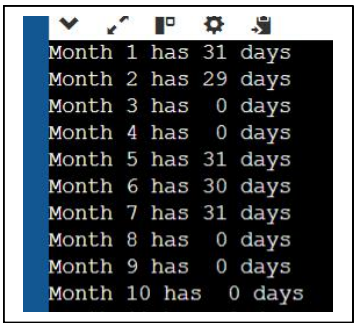
```
   int count=10;
   long sum=0;
   float average=0.0f;

   printf("\n Enter the 10 grades:\n");

   for(int i=0;i<count;++i){
      printf("%2u>",i+1);
      scanf("%d",&grades[i]);
      sum+=grades[i];
   }
   average=(float)sum/count;
   printf("\nAverage of the ten grades entered is : %2f\n",average);

   return 0;

}
```



```
 Enter the 10 grades:
 1>1
 2>2
 3>3
 4>4
 5>9
 6>6
 7>4
 8>3
 9>6
10>34

Average of the ten grades entered is : 7.200000
```

Days in a month(using designated initializers)

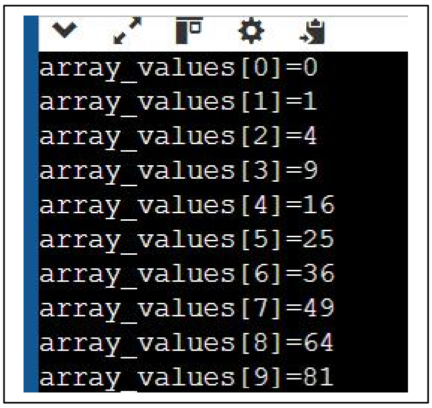
```
#include<stdio.h>
#define MONTHS 12

int main(void){

   int days[MONTHS]={31,28,[4]=31,30,31,[1]=29};
   int index;

   for(index=0;index<MONTHS;index++)
      printf("Month %d has %2d days\n",index+1,days[index]);

   return 0;
}
```



```
Month 1 has 31 days
Month 2 has 29 days
Month 3 has  0 days
Month 4 has  0 days
Month 5 has 31 days
Month 6 has 30 days
Month 7 has 31 days
Month 8 has  0 days
Month 9 has  0 days
Month 10 has  0 days
```

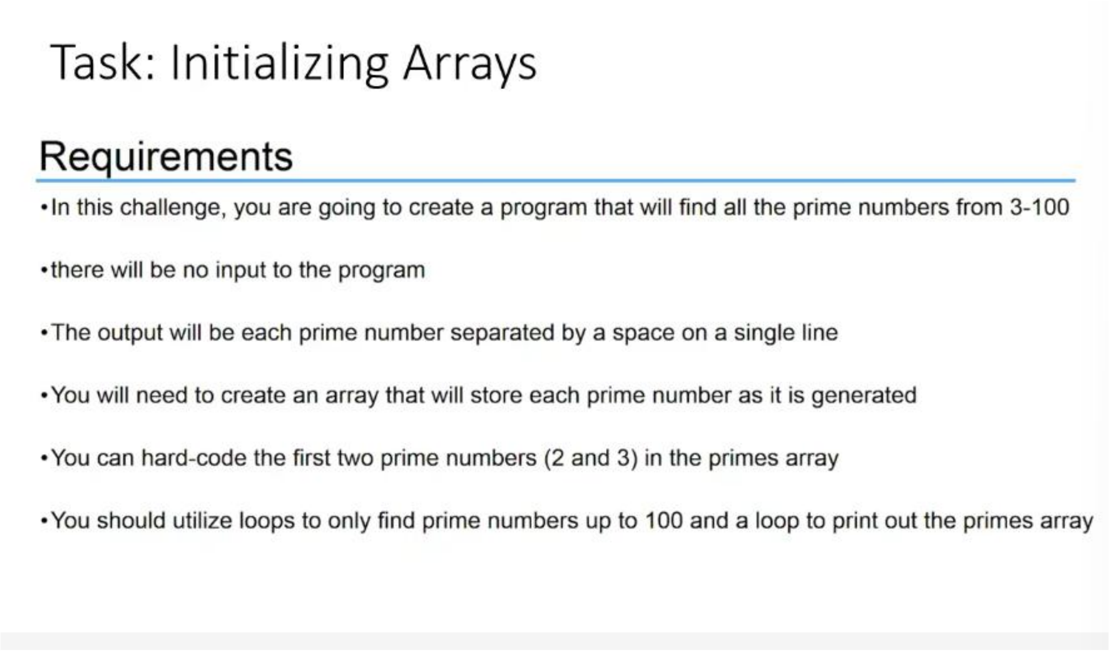
Initializing all elements to the same value

```
#include<stdio.h>

int main(void){
   int array_values[10]={0,1,4,9,16};
   int i;

   for(i=5;i<10;++i)
      array_values[i]=i*i;

   for(i=0;i<10;++i)
      printf("array_values[%i]=%i\n",i,array_values[i]);
```



```
array_values[0]=0
array_values[1]=1
array_values[2]=4
array_values[3]=9
array_values[4]=16
array_values[5]=25
array_values[6]=36
array_values[7]=49
array_values[8]=64
array_values[9]=81
```

```
    return 0;
}
```

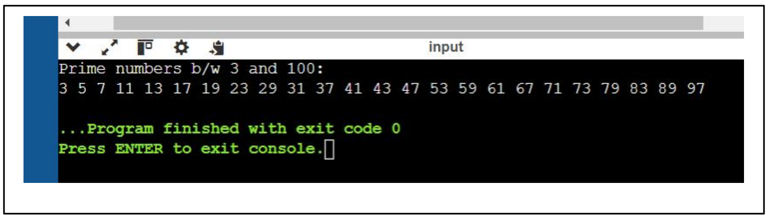


Task: Initializing Arrays

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

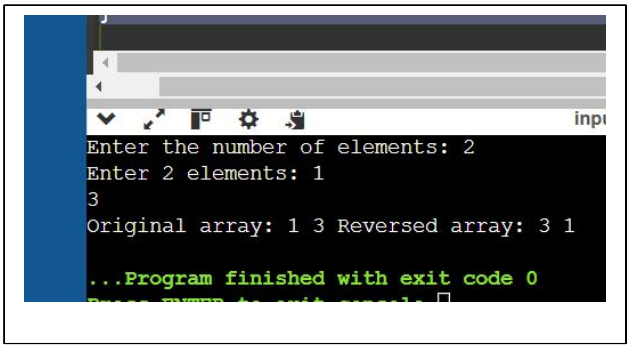
```
#include <stdio.h>
#include <stdbool.h>

int main() {
    int prime_array[100];
    int index = 0;

    for (int i = 3; i <= 100; i++) {
        bool is_Prime = true;

        for (int j = 2; j < i; j++) {
            if (i % j == 0) {
                is_Prime = false;
                break;
            }
        }

        if (is_Prime) {
            prime_array[index] = i;
            index++;
        }
    }

    printf("Prime numbers b/w 3 and 100:\n");
    for (int i = 0; i < index; i++) {
        printf("%d ", prime_array[i]);
```



```
Prime numbers b/w 3 and 100:
3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

...Program finished with exit code 0
Press ENTER to exit console.
```

```
    }

    return 0;
}
```

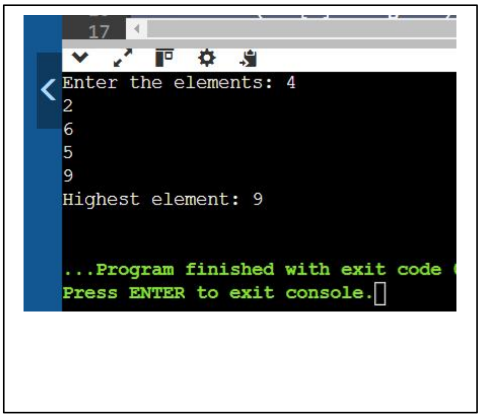
Create a program that reverses the elements of an array. Prompt the user to enter values and print both the original and reversed arrays.



```
Enter the number of elements: 2
Enter 2 elements: 1
3
Original array: 1 3 Reversed array: 3 1

...Program finished with exit code 0
```

```
#include <stdio.h>

int main() {
    int n;
    printf("Enter the number of elements: ");
    scanf("%d", &n);

    int array[n], reverse_array[n];

    printf("Enter %d elements: ", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &array[i]);
    }
    printf("Original array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", array[i]);
    }
    for (int i = 0; i < n; i++) {
        reverse_array[i] = array[n - 1 - i];
    }
    printf("Reversed array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", reverse_array[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.

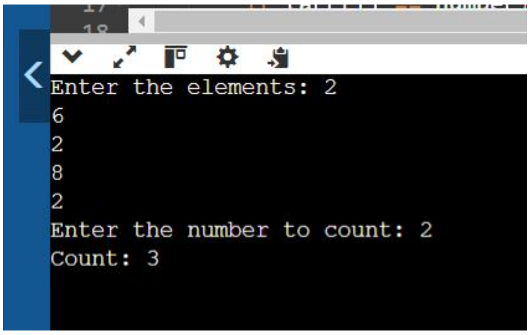


```
Enter the elements: 4
2
6
5
9
Highest element: 9

...Program finished with exit code
Press ENTER to exit console.
```

```
#include <stdio.h>

int main() {
    int arr[5];

    printf("Enter the elements: ");
    for(int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    int highest = arr[0];

    for(int i = 1; i < 5; i++) {
        if (arr[i] > highest) {
            highest = arr[i];
        }
    }

    printf("Highest element: %d\n", highest);

    return 0;
}
```

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

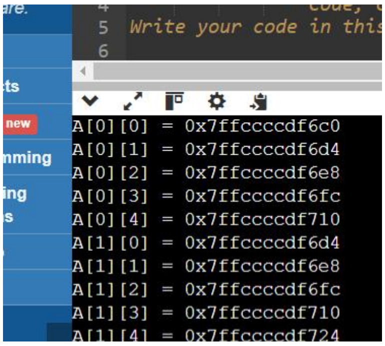
```
#include <stdio.h>

int main() {
    int arr[5], count = 0, number;

    printf("Enter the elements: ");
    for(int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Enter the number to count: ");
    scanf("%d", &number);

    for(int i = 0; i < 5; i++) {
        if (arr[i] == number) {
            count++;
        }
    }

    printf("Count: %d\n", count);

    return 0;
}
```



```
Enter the elements: 2
6
2
8
2
Enter the number to count: 2
Count: 3
```

<u>**MULTIDIMENSIONAL ARRAYS**</u>

```
#include <stdio.h>

int main()
{
```

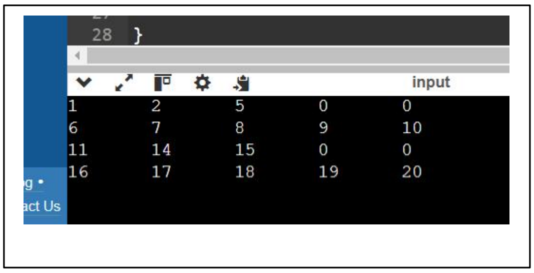


```
Write your code in this
```

```
A[0][0] = 0x7ffccccdf6c0
A[0][1] = 0x7ffccccdf6d4
A[0][2] = 0x7ffccccdf6e8
A[0][3] = 0x7ffccccdf6fc
A[0][4] = 0x7ffccccdf710
A[1][0] = 0x7ffccccdf6d4
A[1][1] = 0x7ffccccdf6e8
A[1][2] = 0x7ffccccdf6fc
A[1][3] = 0x7ffccccdf710
A[1][4] = 0x7ffccccdf724
```

```
    int A[4][5];

    for(int j=0;j<4;j++){
        for(int k=0;k<5;k++){
            printf("A[%d][%d] = %p\n",j,k,(A+j+k));
        }
    }
}
```

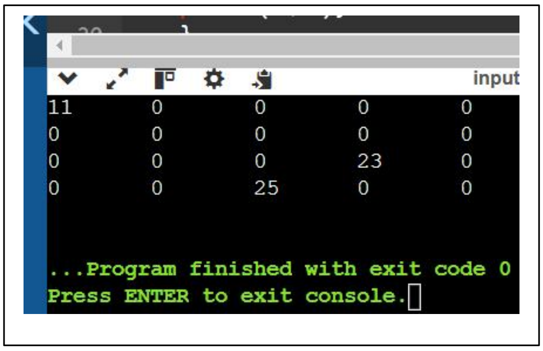
After adding elements



```
1       2       5       0       0
6       7       8       9       10
11      14      15      0       0
16      17      18      19      20
```

```
#include <stdio.h>

int main()
{
    int A[4][5]={
        {1,2,5},
        {6,7,8,9,10},
        {11,14,15},
        {16,17,18,19,20}
    };

    for(int j=0;j<4;j++){
        for(int k=0;k<5;k++){
            printf("%d\t",A[j][k]);
        }
    printf("\n");
    }
}
```

Using Designated Initializers

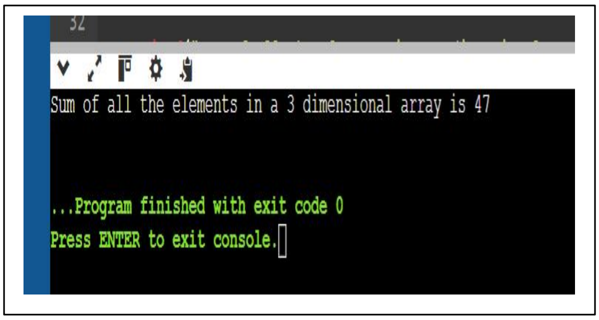


```
11      0       0       0       0
0       0       0       0       0
0       0       0       23      0
0       0       25      0       0


...Program finished with exit code 0
Press ENTER to exit console.
```

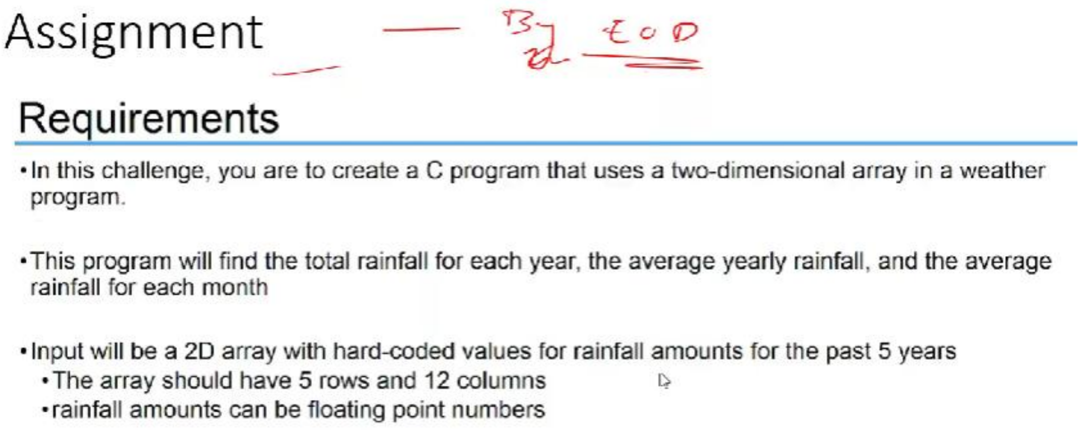
```
#include <stdio.h>

int main()
{
    int A[4][5]={[0][0]=11,[2][3]=23,[3][2]=25};

    for(int j=0;j<4;j++){
        for(int k=0;k<5;k++){
            printf("%d\t",A[j][k]);
        }
    printf("\n");
    }



}
```

3 DIMENSIONAL ARRAY

```
#include <stdio.h>

int main()
{
    int sum=0;
    int num[2][2][2]={
        {
            {1,2},
            {3,4}
        },
        {
            {5,6},
            {7,8}
        }
    };

    for (int i=0;i<2;i++){   //represent no. of stacks
        for(int j=0;j<=2;j++){
            for(int k=0;k<2;k++){
                sum+=num[i][j][k];
            }
        }
    }

    printf("Sum of all the elements in a 3 dimensional array is %d \n",sum);
    return 0;
}
```



```
Sum of all the elements in a 3 dimensional array is 47


...Program finished with exit code 0
Press ENTER to exit console.
```



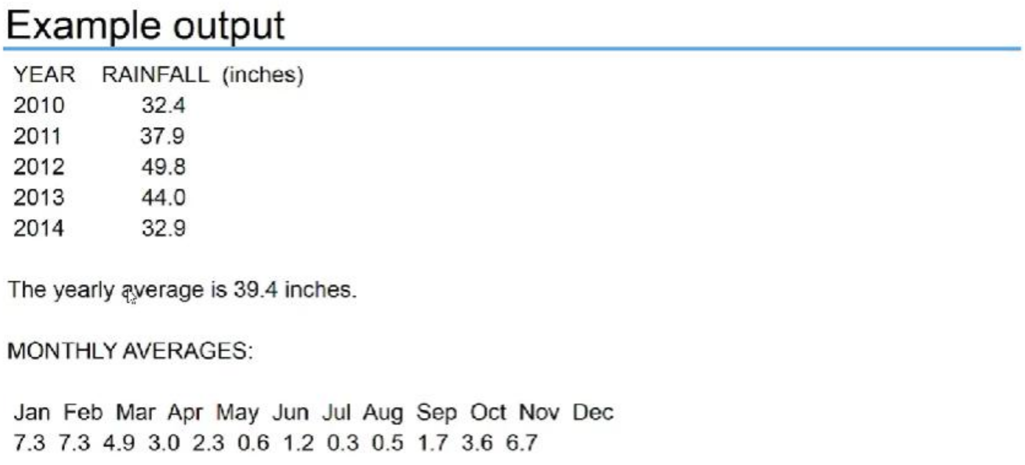
Assignment — By EOD

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

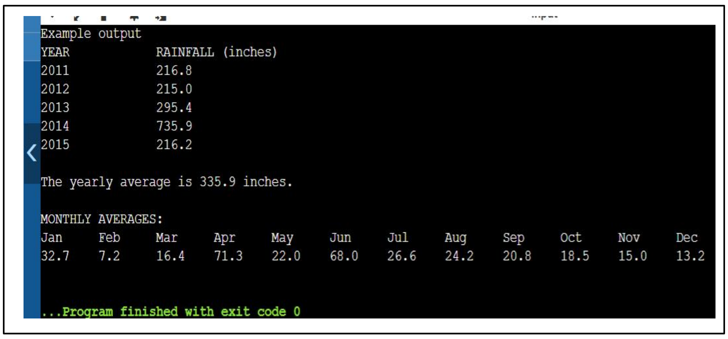
- Input will be a 2D array with hard-coded values for rainfall amounts for the past 5 years
  - The array should have 5 rows and 12 columns
  - rainfall amounts can be floating point numbers



Example output

```
YEAR   RAINFALL  (inches)
2010        32.4
2011        37.9
2012        49.8
2013        44.0
2014        32.9

The yearly average is 39.4 inches.

MONTHLY AVERAGES:

Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
7.3  7.3  4.9  3.0  2.3  0.6  1.2  0.3  0.5  1.7  3.6  6.7
```

```
#include <stdio.h>

int main() {
    float rainfall[5][12] = {
        {10, 11, 18, 18.2, 25, 22, 25, 24, 20.1, 17.8, 14.5, 11.2},
        {11.5, 1, 18, 19.5, 21.8, 24.1, 26.4, 24.7, 21.4, 19.1, 15, 12.5},
        {90.8, 12.1, 15.4, 17.8, 20, 22.4, 24.7, 23,19.7, 17.4, 14.1, 18},
        {32, 10.5, 17.8, 282, 22, 248, 27.1, 25.4, 22.1, 19.8, 16, 13.2},
        {19, 1.2, 13, 18.9, 21.2, 23.5, 29.8, 24.1, 20.8, 18.5, 15.2, 11}
    };

    float total_yearly_rainfall[5] = {0};
    float total_monthly_rainfall[12] = {0};

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

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

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


    printf("Example output\n");
    printf("YEAR\t \tRAINFALL (inches)\n");
```



```
Example output
YEAR            RAINFALL (inches)
2011            216.8
2012            215.0
2013            295.4
2014            735.9
2015            216.2

The yearly average is 335.9 inches.

MONTHLY AVERAGES:
Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct     Nov     Dec
32.7    7.2     16.4    71.3    22.0    68.0    26.6    24.2    20.8    18.5    15.0    13.2


...Program finished with exit code 0
```

```
    for (int i = 0; i < 5; i++) {
        printf("201%d\t \t%.1f\n", i + 1, total_yearly_rainfall[i]);
    }

    printf("\nThe yearly average is %.1f inches.\n\n", average_yearly_rainfall);

    printf("MONTHLY AVERAGES:\n");
    printf("Jan\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]);
    }

    printf("\n");

    return 0;
}
```