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.

```c
#include<stdio.h>
float  const pi=3.14;
int main()
{
    printf("%0.2f",pi);
    pi=3.15;
    printf("%0.2f",pi);
    float *ptr;
    ptr=&pi;
    *ptr=3.16;
}
```

Output

```
prog50.c: In function 'main':
prog50.c:6:7: error: assignment of read-only variable 'pi'
    pi=3.15;
```

==========

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>
#include<stdint.h>
int main()
{
    uint8_t  a=80,b=70;
    uint8_t const *ptr=&a;
    printf("%p %d \n",ptr,*ptr);
    ptr=&b;
    printf("%p %d",ptr,*ptr);
    *ptr=90;
    printf("%p %d",ptr,*ptr);
}
```

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>
#include<stdint.h>
int main()
{
    uint8_t  a=80,b=70;
    uint8_t *const ptr=&a;
    printf("%p %d \n",ptr,*ptr);
    ptr=&b;
    printf("%p %d",ptr,*ptr);
    *ptr=90;
    printf("%p %d",ptr,*ptr);
}
```

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.

```c
#include<stdio.h>
#include<stdint.h>
int main()
{
    uint8_t  a=80,b=70;
    uint8_t   const *const ptr=&a;
    printf("%p %d \n",ptr,*ptr);
    ptr=&b;
    printf("%p %d",ptr,*ptr);
    *ptr=90;
    printf("%d",a);
}
```

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.

```c
#include<stdio.h>
#include<stdint.h>
void func(int a)
{
    printf("value of a is %d",a);
}
int const b=3;
int main()
{
    func(b);
}
```

Output

value of a is 3

==========

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.

```c
#include<stdio.h>

#include<stdint.h>
int main()
{
 char const arr[7][10]={"sunday", "monday", "tuesday", "wednesday", "thursday",
"friday", "saturday"};
 for(int i=0;i<7;i++)
 {
    printf("%s ",arr[i]);
     printf("\n");
 }
```

```
}
```

Output
sunday
monday
tuesday
wednesday
thursday
friday
Saturday
==========

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.

```c
#include<stdio.h>
#include<stdint.h>
float const pi=3.14;
int main()
{
    int rad;
    float area;
    printf("enter the radius of the circle \n");
    scanf("%d",&rad);
    area=pi*rad*rad;
    printf("the area of the circle is %f",area);
}
```

Output

enter the radius of the circle
3
the area of the circle is 28.260000

==========

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.

```c
#include<stdio.h>
#include<stdint.h>
int const val=10;
int main()
{
    for(int i=0;i<val;i++)
    {
        printf("%d ",i);
    }
}
```

Output

0 1 2 3 4 5 6 7 8 9

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.

```c
#include<stdio.h>
#include<stdint.h>
int const b=3;
void func(int a)
{
    printf("value of a is %d",a);
    printf("%d",b);
}
int main()
{
    int q=1;
    func(q);
}
```

10. Save the prime numbers between 3 and 100 in an array and print the array.

```c
#include<stdio.h>
int main()
{
    int arr[25]={2,3},count=0,i=2;

        for(int j=4;j<100;j++)
        {
```

```
            for(int k=2;k<=j/2;k++)
            {
                if(j%k==0)
                    count++;
            }
            if(!count)
                arr[i++]=j;
            count=0;
        }

        for(int i=0;i<25;i++)
        {
            printf("%d ",arr[i]);
        }
}
```

Output

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. 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 arr[10],n;
   printf("enter the number of elements of the array \n");
   scanf("%d",&n);
   printf("enter the elements if the array \n");
   for(int i=0;i<n;i++)
   {
       scanf("%d",&arr[i]);
   }
   printf("the elements of the array are \n");
   for(int i=0;i<n;i++)
   {
       printf("%d ",arr[i]);
   }
    printf("the elements of the reversed array are \n");
     for(int i=n-1;i>=0;i--)
   {
       printf("%d ",arr[i]);
```

```
    }
 }
```

Output

enter the number of elements of the array
5
enter the elements if the array
1 3 5 7 9
the elements of the array are
1 3 5 7 9
the elements of the reversed array are
9 7 5 3 1

---

12. 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 arr[10],n,highest=0;
    printf("enter the number of elements of the array \n");
    scanf("%d",&n);
    printf("enter the elements of the array \n");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    for(int i=0;i<n;i++)
    {
        if(arr[i]>highest)
        {
            highest=arr[i]
        }
    }
        printf("highest  element of the array is %d \n",highest);
 }
```

Output

enter the number of elements of the array
5
enter the elements of the array
1 3 4 5 6
highest element of the array is 6

---

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

```c
#include<stdio.h>
 int main()
 {
    int arr[10],n,num,count=0;
    printf("enter the number of elements of the array \n");
    scanf("%d",&n);
    printf("enter the elements of the array \n");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("enter the element to be counted \n");
    scanf("%d",&num);
     for(int i=0;i<n;i++)
    {
        if(num==arr[i])
        {
            count++;
        }
    }
    printf("%d occured %d times \n",num,count);
 }
```

Output

enter the number of elements of the array
10
enter the elements of the array
3 4 3 2 6 3 8 3 1 1
enter the element to be counted
3
3 occured 4 times