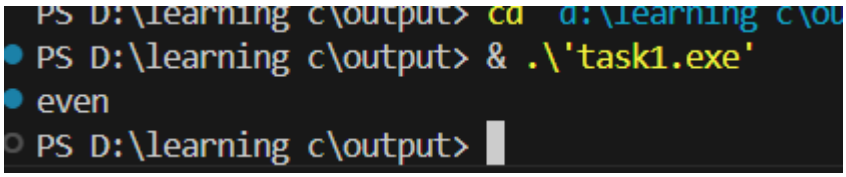
ASSIGNMENT

1.Write a C program to determine if the least significant bit of a given integer is set (i.e., check if the number is odd).

```
#include<stdio.h>

int main()

{

   int number=50;

   if(number|1)

   {

      printf("even");

   }

   else

   {

      printf("odd");

   }


}
```



2. Create a C program that retrieves the value of the nth bit from a given integer.

```
// c program to retrieve value of nth bit from a given integer

#include<stdio.h>

int main()

{

   int number;

   printf("enter a number");

   scanf("%d",&number);
```

```
    int n;
    printf("enter the bit");
    scanf("%d",&n);
    int result;
    result=number&1<<n;
    if(result)
    {
        printf("the n th bit is 1");
    }
    else
    {
        printf("the nth bit is zero");
    }


}
```

```
PS D:\learning c\output> & .\'task2.exe'
enter a number10
enter the bit0
the nth bit is zero
PS D:\learning c\output> cd 'd:\learning c\output'
PS D:\learning c\output> & .\'task2.exe'
enter a number2
enter the bit1
the n th bit is 1
PS D:\learning c\output>
```

3. Develop a C program that sets the nth bit of a given integer to 1.

```
#include<stdio.h>
int main()
{
    int n,num,result;
```

```
    printf("enter a number");
    scanf("%d",&num);
    printf("enter the bit which you want to set to 1");
    scanf("%d",&n);
    result=num|1<<n;
    printf("result is %d",result);

}
```

```
PS D:\learning c\output> & .\'task3.exe'
enter a number10
enter the bit which you want to set to 10
result is 11
PS D:\learning c\output>
```

4. Write a C program that clears (sets to 0) the nth bit of a given integer.

```
// c program that clears the nth bit of given integer
#include<stdio.h>
int main()
{

    int n,num,result,mask;
    printf("enter a number");
    scanf("%d",&num);
    printf("enter the bit which you want to clear");
    scanf("%d",&n);
    mask=(1<<n);
    printf("mask is %d",mask);
    int mask2=~mask;
    result=num&mask2;
    printf("result is %d",result);

}
```

```
PS D:\learning c\output> & .\'task4.exe'
enter a number10
enter the bit which you want to clear1
mask is 2result is 8
PS D:\learning c\output>
```

5. Create a C program that toggles the nth bit of a given integer.

```
// c program that toggles the nth bit of a given integer
#include<stdio.h>
int main()
{
   int n,num,result;
   printf("enter a number");
   scanf("%d",&num);
   printf("enter the bit which you want to set to 1");
   scanf("%d",&n);
   result=num^1<<n;
   printf("result is %d",result);

}
```

```
PS D:\learning c\output> & .\'task5.exe'
enter a number8
enter the bit which you want to set to 11
result is 10
PS D:\learning c\output>
```

6. Write a C program that takes an integer input and multiplies it by 2^n using the left shift operator.

```
#include<stdio.h>
int main()
{
   int num,n;
```

```
    printf("enter a number");
    scanf("%d",&num);
    printf("enter n ");
    scanf("%d",&n);
    int result=num<<n;
    printf("result is %d",result);

}
```

```
PS D:\learning c\output> & .\'task6.exe'
enter a number24
enter n 1
result is 48
PS D:\learning c\output>
```

7. Create a C program that counts how many times you can left shift a number before it overflows (exceeds the maximum value for an integer).

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

int main() {
    int number = 1;
    int shift_count = 0;

    while (number > 0 && number <= INT_MAX / 2) {
        number <<= 1;
        shift_count++;
    }

    printf("You can left shift the number 1 a maximum of %d times before overflow occurs.\n", shift_count);

    return 0;
}
```

```
PS D:\learning c\output> & .\'task7.exe'
You can left shift the number 1 a maximum of 30 times before overflow occurs.
PS D:\learning c\output>
```

8. Write a C program that creates a bitmask with the first n bits set to 1 using the left shift operator.

```
#include<stdio.h>
int main()
{
   int n ,mask=0;
   printf("enter the n bits");
   scanf("%d",&n);
   for(int i=0;i<n;i++)
   {
      mask = mask|(1<<i);
   }
   printf("the mask is %d",mask);



}
```

```
PS D:\learning c\output> & .\'task8.exe'
enter the n bits4
the mask is 15
PS D:\learning c\output>
```

9. Develop a C program that reverses the bits of an integer using left shift and right shift operations.

```
#include<stdio.h>
int main()
{
   int num,result;
```

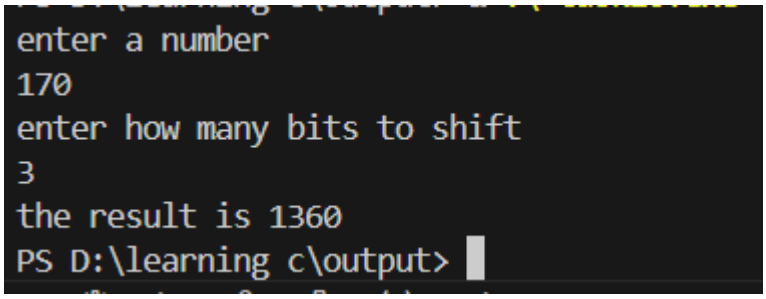
```
    printf("enter a number");
    scanf("%d",&num);
    int bitcount=8;
    for(int i=0;i<bitcount;i++)
    {
       result=num<<i;



    }
    printf("number is %d",result);

}
```

```
enter a number5
number is 640
PS D:\learning c\output> cd 'd:\learning c\output'
PS D:\learning c\output> & .\'task9.exe'
enter a number10
number is 1280
PS D:\learning c\output> cd 'd:\learning c\output'
PS D:\learning c\output> & .\'task9.exe'
enter a number15
number is 1920
```

10. Create a C program that performs a circular left shift on an integer.

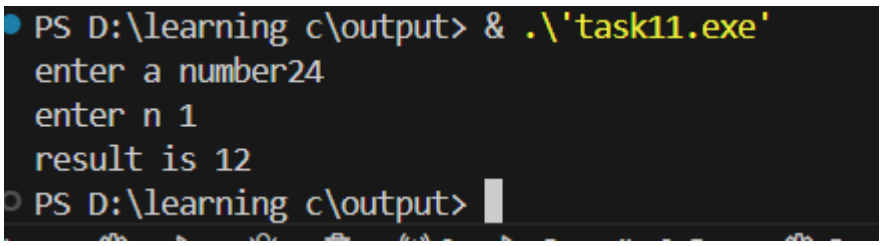
```
#include<stdio.h>
int main()
{
    int num,result,n;
    printf("enter a number \n");
    scanf("%d",&num);
    int bitcount=32;
    printf("enter how many bits to shift \n");
    scanf("%d",&n);
```

```
    result=num<<n | num>>(bitcount-n);
    printf("the result is %d",result);

}
```



11. Write a C program that takes an integer input and divides it by 2^ n using the right shift operator.

```
#include<stdio.h>
int main()
{
    int num,n;
    printf("enter a number");
    scanf("%d",&num);
    printf("enter n ");
    scanf("%d",&n);
    int result=num>>n;
    printf("result is %d",result);

}
```



12. Create a C program that counts how many times you can right shift a number before it becomes zero.

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

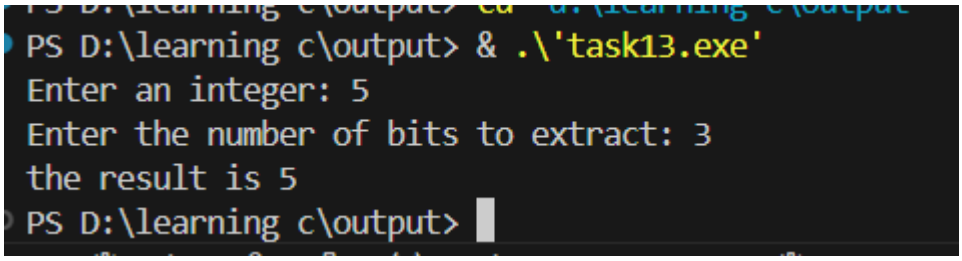
```
{
   int num;
   printf("enter a number");
   scanf("%d",&num);
   int count=0;
   while(num>0)
   {
      num=num>>1;
      count=count+1;

   }
   printf("the count is %d",count);
}
```

```
PS D:\learning c\output> & .\'task12.exe'
enter a number24
the count is 5
PS D:\learning c\output> cd 'd:\learning c\ou
PS D:\learning c\output> & .\'task12.exe'
enter a number24
the count is 5
PS D:\learning c\output>
```

13. Write a C program that extracts the last n bits from a given integer using the right shift operator.

```
#include <stdio.h>

int main() {
   int num, n;


   printf("Enter an integer: ");
   scanf("%d", &num);
```

```
    printf("Enter the number of bits to extract: ");
    scanf("%d", &n);



    int bits = num & ((1 << n) - 1);


    // Output the result
    printf("the result is %d", bits);


    return 0;
}
```



14. Develop a C program that uses the right shift operator to create a bitmask that checks if specific bits are set in an integer.

```
#include<stdio.h>
int main()
{
    int num=10;
    int n;
    printf("enter the bits to set");
    scanf("%d",&n);
    int mask=1<<n;
    printf("mask is %d",mask);


    int result;
    result=num|mask;
```

```
    printf("the result is %d",result);


}
```

```
PS D:\learning c\output> & .\'task14.exe'
enter the bits to set0
mask is 1the result is 11
PS D:\learning c\output>
```

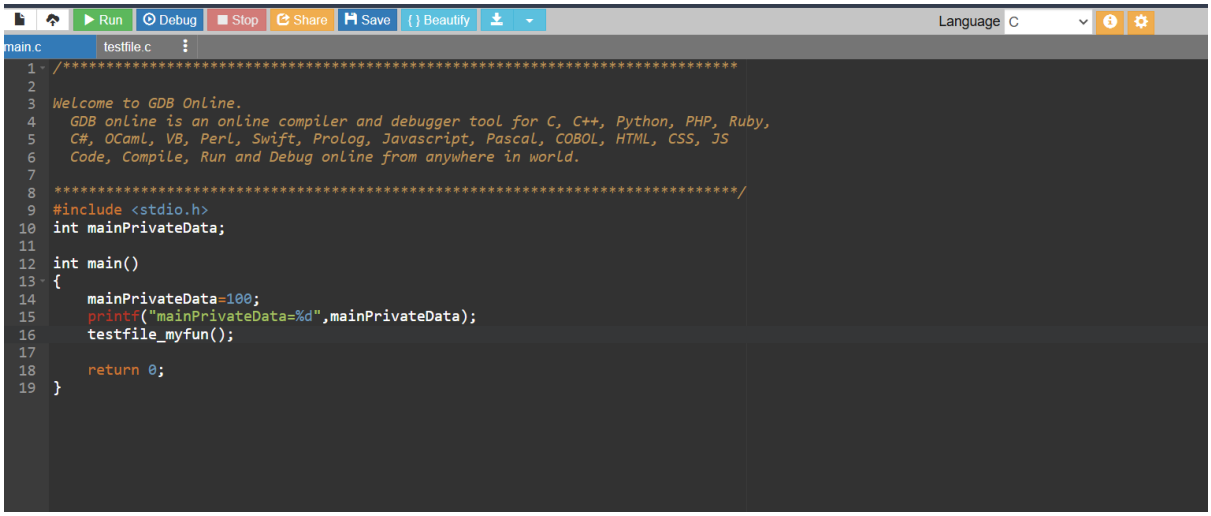
CLASS WORK

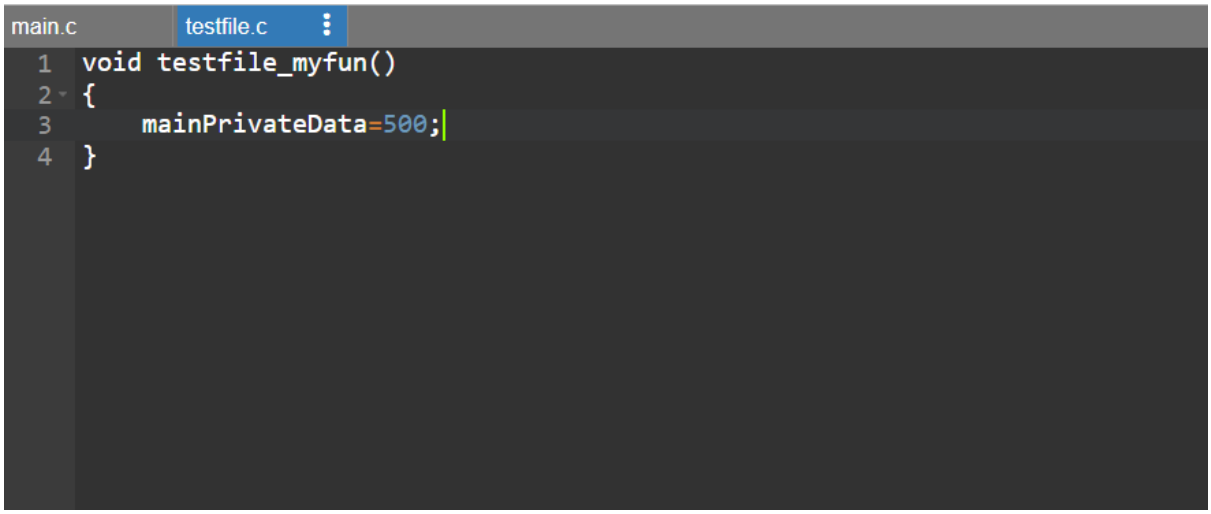
1.
```
#include<stdio.h>
void myFun(void);
//int count=0;//autostorage class but changed the scope
static int count=0;
int main()
{


    int p; //storage class =auto(by default)
    myFun();
    myFun();
    myFun();
    myFun();
    printf("002the function is executed %d times \n",count);


}
void myFun()
{
   // int count=0;//auto storage class
    count=count+1;
    printf("001the function is executed %d times \n",count);
}
```

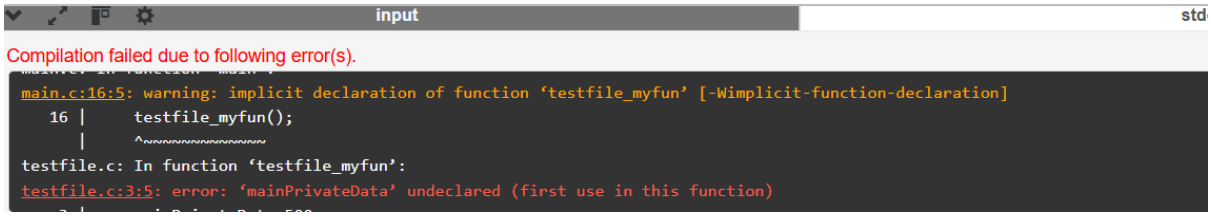
2.
```
/* requirement:we want a global variable that is private
to a functio. we want a private variable that doesnot
```

```
loose its existence even if the execution control
goes out of the scope of that variable*/
#include<stdio.h>
void myFun(void);
int main()
{



   myFun();
   myFun();
   myFun();
   myFun();



}
void myFun()
{
  static int count=0;
   count=count+1;
   printf("001the function is executed %d times \n",count);
}
```

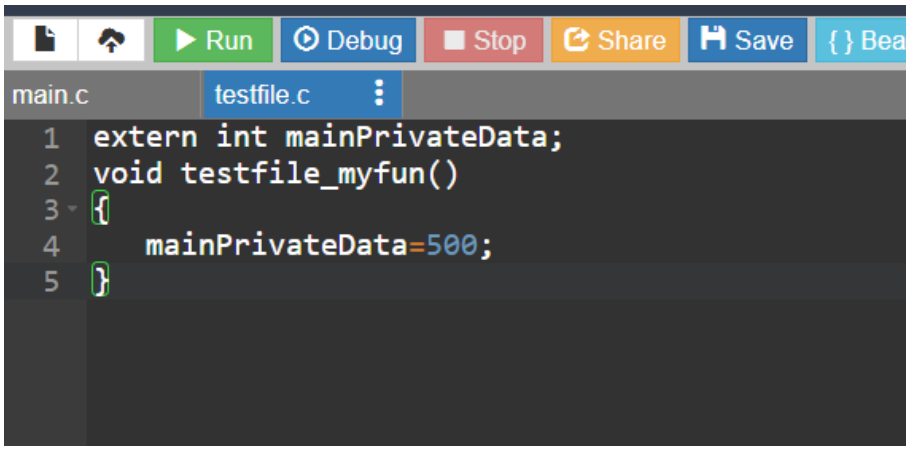
3.







How to correct:



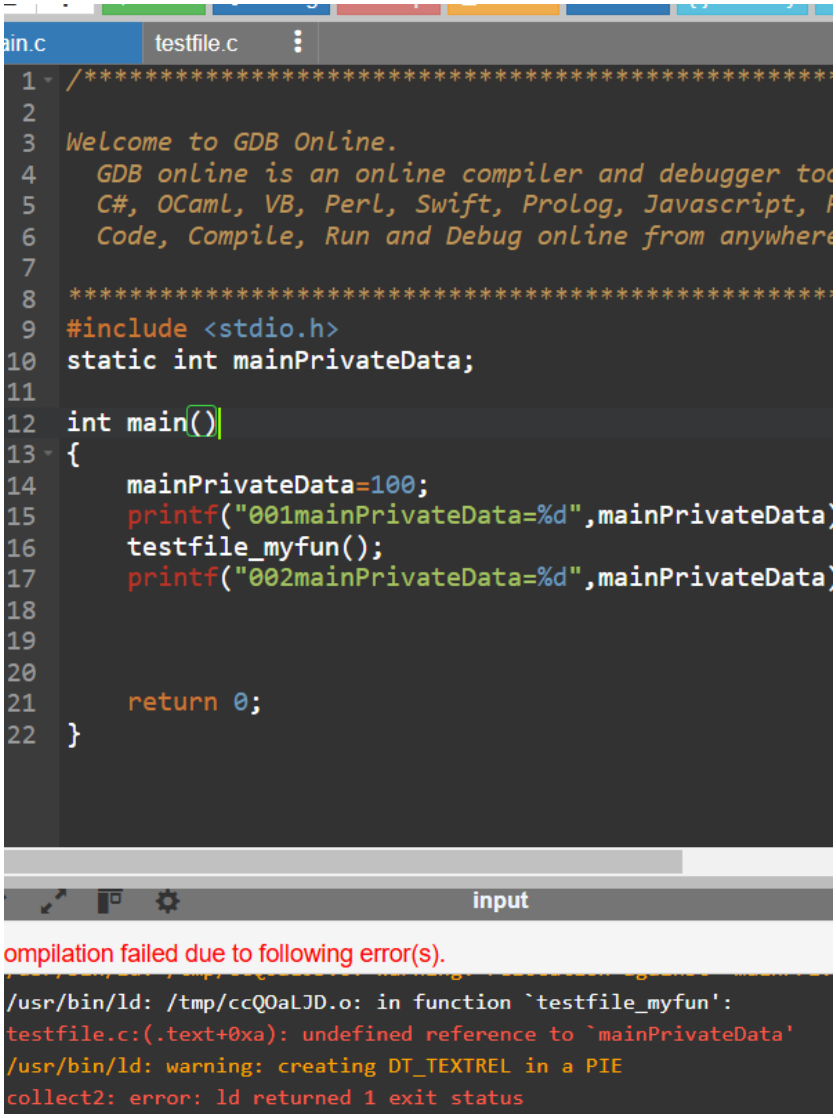
```
fun' [-Wimplicit-function-declaration]
   16 |     testfile_myfun();
      |     ^~~~~~~~~~~~~~~
mainPrivateData=100
```

main.c testfile.c

```c
/******************************************************************

Welcome to GDB Online.
  GDB online is an online compiler and debugger tool for C, C++,
  C#, OCaml, VB, Perl, Swift, Prolog, Javascript, Pascal, COBOL,
  Code, Compile, Run and Debug online from anywhere in world.

*******************************************************************/
#include <stdio.h>
int mainPrivateData;

int main()
{
    mainPrivateData=100;
    printf("001mainPrivateData=%d",mainPrivateData);
    testfile_myfun();
    printf("002mainPrivateData=%d",mainPrivateData);



    return 0;
}
```

input

```
myfun' [-Wimplicit-function-declaration]
   16 |     testfile_myfun();
      |     ^~~~~~~~~~~~~~~
001mainPrivateData=100002mainPrivateData=500
```



The variable visibility is limited to main.c only here , if we use static .

4.

main.c testfile.c

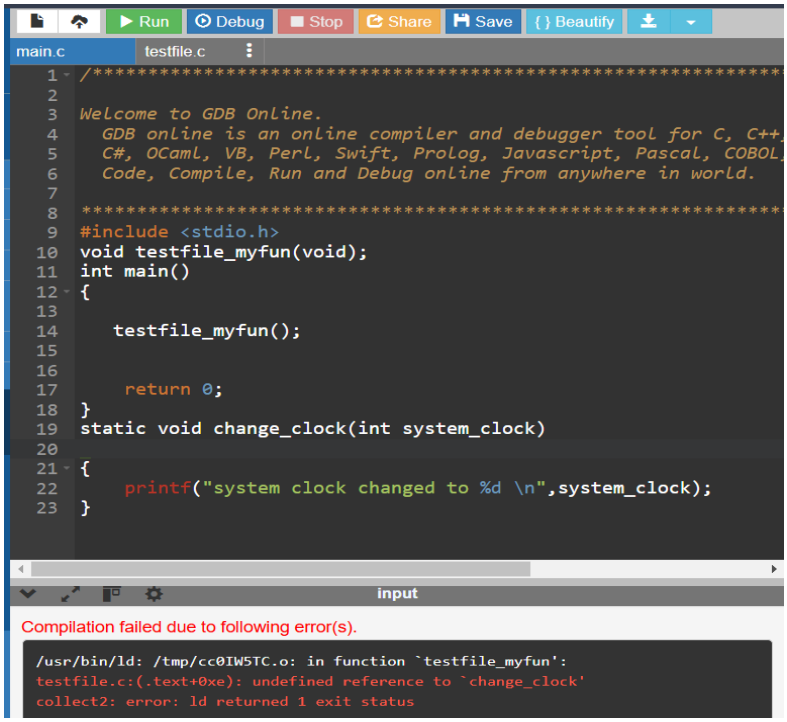
```
/*****************************************************************

Welcome to GDB Online.
  GDB online is an online compiler and debugger tool for C, C++,
  C#, OCaml, VB, Perl, Swift, Prolog, Javascript, Pascal, COBOL,
  Code, Compile, Run and Debug online from anywhere in world.

*****************************************************************/
#include <stdio.h>
void testfile_myfun(void);
int main()
{

   testfile_myfun();


    return 0;
}
void change_clock(int system_clock)
{
    printf("system clock changed to %d \n",system_clock);
}
```

main.c testfile.c

```
extern int mainPrivateData;
extern void change_clock(int);
void testfile_myfun()
{
    change_clock(50);

}
```

input

```
system clock changed to 50


Program finished with exit code 0
```



5. #include<stdio.h>

int main()

{

char a=40;

char b=30;

printf("& operation %d \n",a&b);

printf("|operation %d \n",a|b);

printf("^ operation %d \n",a^b);

printf("~ operation %d \n",~a);

}

6. #include<stdio.h>

int main()

{

int a=10,b=4;

int c=a>b;

printf("001 c= %d \n ",c);

c=a<b;

```
    printf("002 c =%d \n",c);
    c=a>=b;
    printf("003 c = %d \n",c);
    c=a<=b;
    printf("004 c = %d \n",c);
    c=a==b;
    printf("005 c=%d \n",c);
    c=a!=b;
    printf("006 c=%d \n",c);
}
```