//FILE INPUT

```
/*Read Values from the Input file*/
#include <stdio.h>
int main()
{
//file pointer definition to hold a disk location
   FILE * fp;
   fp = fopen("inputFile.txt","r");
//assign its address/disk location to file pointer
   char firstWord[20];
   char secondWord[20];
   int num;
   if(fp==NULL)
   {
      printf("File doesnot Exist...");
      return 1;
   }
   printf("Reads two words and an integer from file \n");
   fscanf(fp,"%s %s %d", firstWord,secondWord,&num);

   printf("Display back what has been read from input file: \n");
    printf("%s %s \n%d \n",firstWord,secondWord,num);
    fclose(fp);
    return 0;
}
```

OUTPUT: File Input

```
Reads two words and an integer from file
Display back what has been read from input file:
Hello World
4
PS C:\TBC\Sem2\Principle of programming\POP\Weekly
```

fileInput.c ● inputFile.txt ×

Weekly_Assignment > Practical3 > inputFile.

```
1   Hello World
2   4
```

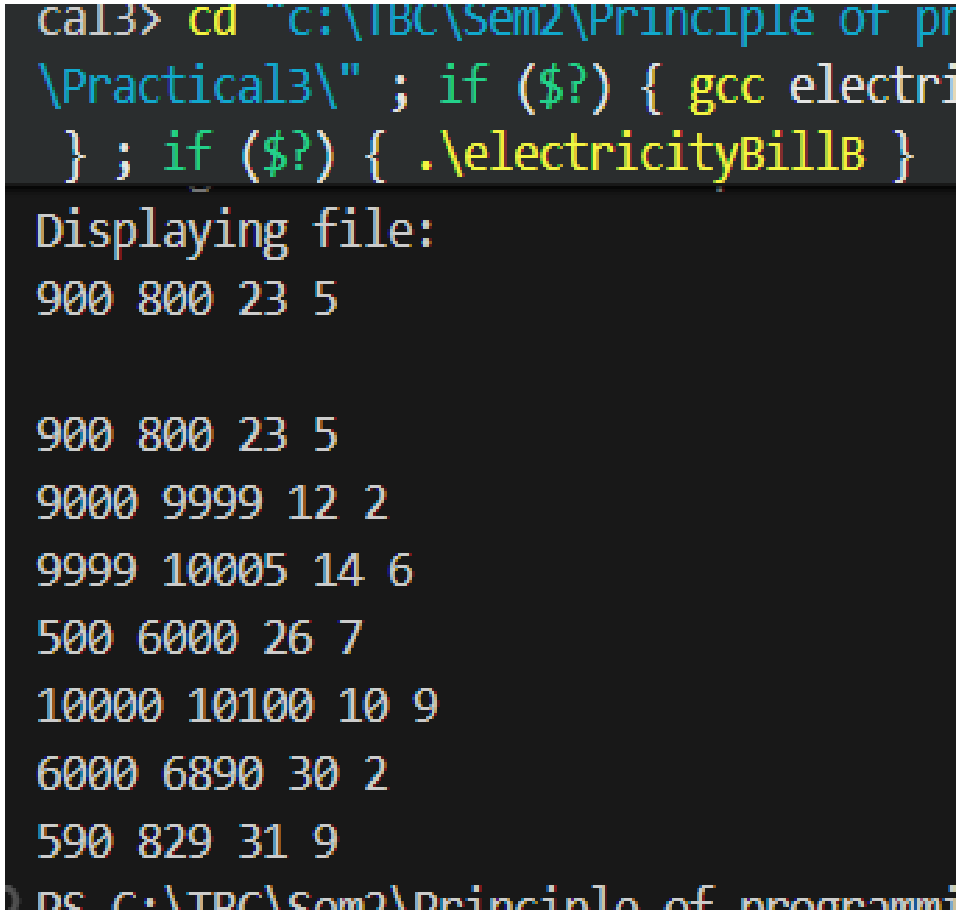
What are the differences between these two blocks of code?

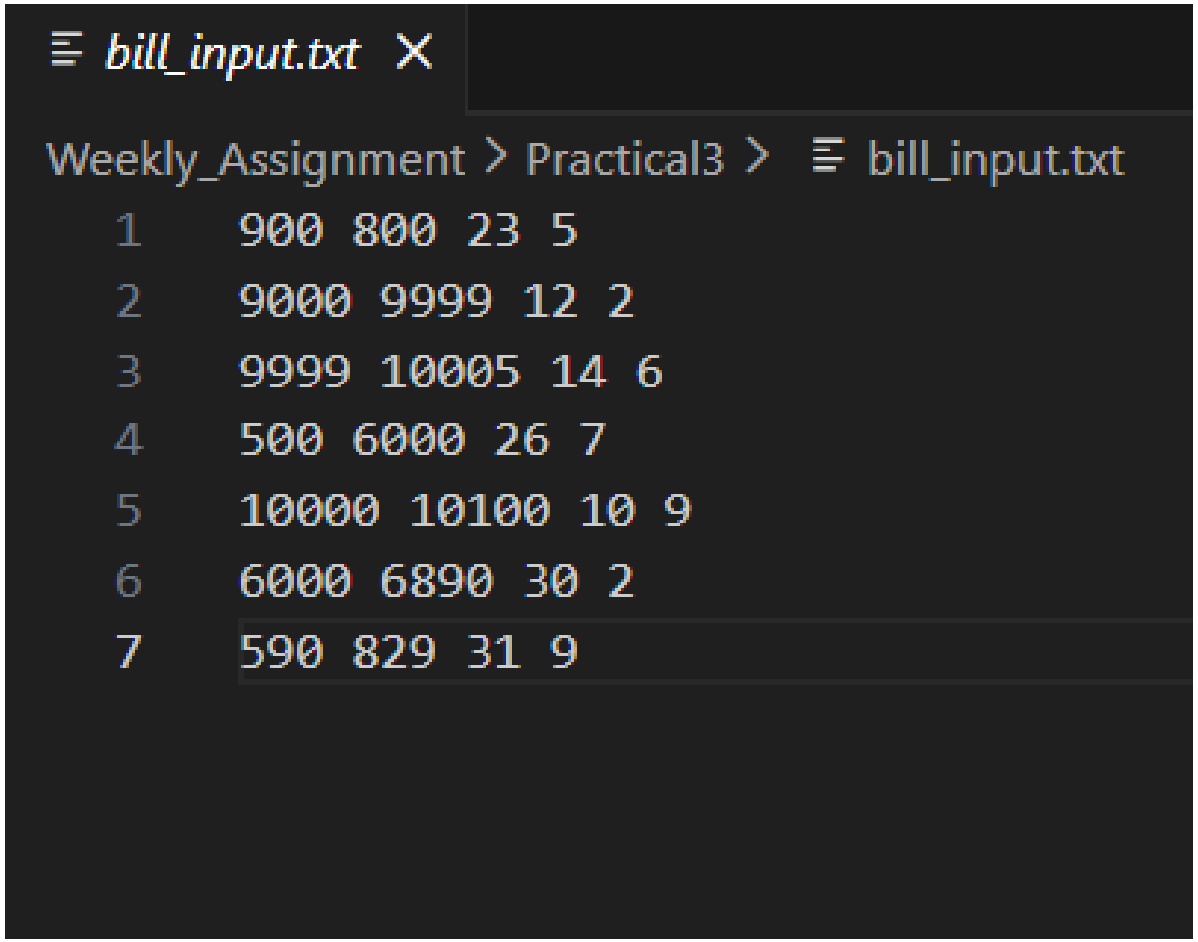
: The fscanf() function takes input from the txt file

: The scanf() function takes input from the users.

```
//READ ELECTRICITY BILL
#include <stdio.h>


int main()
{
   FILE * fp;
   int a,b,c,d;
   fp = fopen("bill_input.txt","r");

   if(fp==NULL)
   {
      printf("File doesnot Exist...");
      return 1;
   }
   printf("Reading Values from bill input \n");
   fscanf(fp,"%d %d %d %d",&a,&b,&c,&d);
   printf("Displaying file: \n");
   printf("%d %d %d %d\n\n",a,b,c,d);


   // Goes (starts) to the beginning of the file
   fseek(fp,0,SEEK_SET);
   for (int i=1;i<=7;i++)
   {
      fscanf(fp,"%d %d %d %d",&a,&b,&c,&d);
      printf("%d %d %d %d\n",a,b,c,d);
   }


   fclose(fp);
   return 0;
}
```

OUTPUT: READ ELECTRICITY BILL





//Customer Statistics Pseudo Code

/*

Step 1: Start

Step 2: Open file

Step 3: Take input from the file

Step 4: If about 1000 heave user,

    else if 500-100 unit consumed regular user

    below 0-500 unit consumed light user

    else, should enter [positive integer(more than 0)

Step 5: Count the unit consumer

Step 6: Display the number of consumer, classifiying the users

Step 7: End

*/

```
//Customer Statistics
#include <stdio.h>


int main()
{
   FILE * fp;
   int unit,test_int;
   int count_huser=0, count_ruser=0, count_luser=0;
   fp = fopen("testInput.txt","r");


   if(fp==NULL)
   {
      printf("File doesnot Exist...");
      return 1;
   }
   //takes input from the file unit it reaches the end of file
```

```
while((test_int=fscanf(fp,"%d",&unit))!=EOF)
 {
      if (unit>=1000)
      {
         count_huser+=1;
      }
      else if (unit>=500 && unit<1000)
      {
         count_ruser+=1;
      }
      else if (unit>0 && unit<500)
      {
         count_luser+=1;
      }
      else
      {
         printf("Invalid Input!!");
      }
 }
 printf("\nHeavy Users :%d\n",count_huser);
 printf("Regular Users :%d\n",count_ruser);
 printf("Light Users :%d\n",count_luser);

 fclose(fp);
 return 0;
}
```

Output: Customer Statistics

```
PS C:\TBC\Sem2\Principle of programmi
cal3> cd "c:\TBC\Sem2\Principle of pr
\Practical3\" ; if ($?) { gcc custome
tics } ; if ($?) { .\customerStatisti

Heavy Users :12
Regular Users :3
Light Users :1
PS C:\TBC\Sem2\Principle of programmi
```

testInput.txt

Weekly_Assignment > Practical3 > testInput.txt

```
1   5
2   1078 9000 9999
3   1234 10000 10100
4   1134 6000 6890
5   4511 590 829
6   8122 500 2000
```

Display Pattern:

```
#include <stdio.h>


//PATTERNS
int main ()
{
    int size = 6;
    //using temp as copy variable of size to not affect the actual size variable
    int temp = size;
    int temp2 = size;
```

to be continued...

Display Pattern: Pattern A

```
printf("Patter A: \n");
for(int i=1;i<=size;i++)
{
   for (int j =1;j<=i;j++)
   {
      printf("%d ",j);
   }

   printf("\n");
}
printf("\n\n");
```

- OUTPUT:

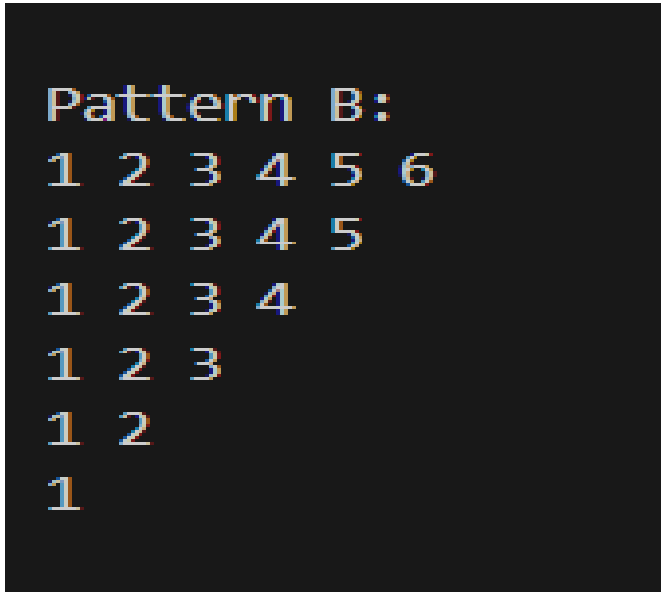
```
PS cd "c:\TBC\Sem2\Principle of programm
displayPatterns.c -o displayPatterns } ;
Patter A:
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
```

to be continued...

Display Pattern: Pattern B

```
printf("Pattern B: \n");
for(int i=1;i<=size;i++)
{

    for(int j =1; j<=temp;j++)
    {
        printf("%d ",j);
    }
    temp-=1;

    printf("\n");
}
printf("\n\n");
```

- OUTPUT:

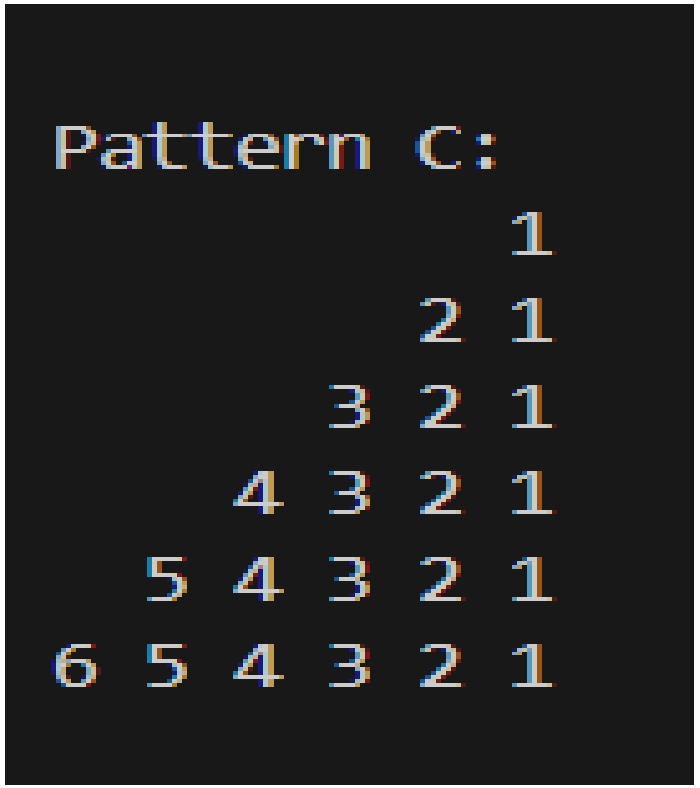


to be continued...

Display Pattern: Pattern C

```
printf("Pattern C: \n");
for(int i=1;i<=size;i++)
{
   for(int space =i;space<size;space++)
   {
      printf("  ");
   }
   //reversing the pattern
   for(int j=i; j>=1;j--)
   {
      printf("%d ",j);
   }
   //temp2-=1;
   printf("\n");
}
printf("\n\n");
```

- OUTPUT:

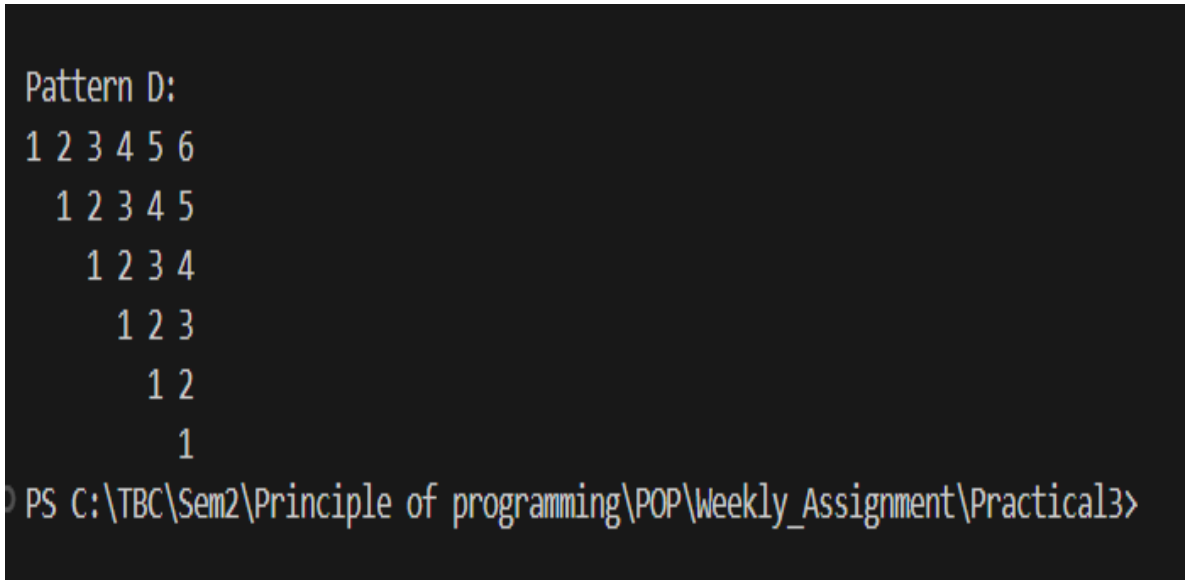


to be continued...

Display Pattern: Pattern D

```
  printf("Pattern D: \n");
   for(int i=1;i<=size;i++)
   {
      for(int space=1;space<i;space++)
      {
         printf("  ");
      }
      for(int j = 1;j<=temp2;j++)
      {
         printf("%d ",j);
      }
      temp2-=1;
      printf("\n");
   }
   return 0;
}
```

- OUTPUT:



//Largest num, count

```
#include <stdio.h>


int main()
{
   FILE * fp;
   fp = fopen("occurrence.txt","r");
   int size = 10,num[size], largest=num[0];
   int l_occurrence =0;


   if (fp==NULL)
   {
      printf("File doesnot exist");
      return 1;
   }
```

```
//Reading values from file
for(int i =0;i<size;i++)
{
   fscanf(fp,"%d",&num[i]);
   //Assuming  input ends with 0
   //breaks if input is 0 while fetching
   if(num[i]==0)
   {
      break;
   }
   if (largest<num[i])
   {
      largest = num[i];
   }
}
```

to be continued...

... Largest num, count (continued)

```
//counting the occurrence of largest number
for(int j = 0; j<size;j++)
{
   if(num[j]==0)
   {
      break;
   }
   if(largest== num[j])
   {
      l_occurrence+=1;
   }
}
   printf("Largest number: %d\n",largest);
   printf("Largest number occurrence: %d\n",l_occurrence);
   fclose(fp);
   return 0;
}
```

- OUTPUT:

```
($?) { .\exercise3_2 }
Largest number: 5
Largest number occurrence: 4
PS C:\TBC\Sem2\Principle of programmi
```

Vowel and Consonant Count

```
#include <stdio.h>
int main()
{
   int t_vowel=0, t_consonants=0;
   char str[100];
   // To compare sentence with vowel letter
   char vowel[10]= {'a','e','i','o','u','A','E','I','O','U'};
   printf("Enter a word : ");

   // Takes (reads) input until user enters new line.
   scanf("%[^\n]",str);

   /*loops runs if the string is not null char
   as string ends with null character when it reaches
   last array(null char) loop ends*/
   for (int i =0; str[i]!='\0';i++)
   {

      //checks whether the given input is alphabet or not
      //done so to not count space in total consonant
      if(str[i]>='A' && str[i]<='Z' || str[i]>='a' &&
str[i]<='z')
      {
         // the given input is not vowel, resets after each
letter
         int is_vowel =0;
         for(int j =0; j<10;j++)
         {
            if(str[i]==vowel[j])
            {
               // if the given input is vowel
               is_vowel =1;
            }
         }
```

to be continued...

… Vowel and Consonant Count continued

```
        if (is_vowel==1)
         {
            t_vowel+=1;
         }
         if (is_vowel == 0)
         {
            t_consonants+=1;
         }
      }
   }

   printf("\nTotal Vowel: %s",str);
   printf("\nTotal Vowel: %d\n",t_vowel);
   printf("Total Consonants: %d",t_consonants);
   return 0;
}
```

- OUTPUT: