Simulate all File allocation strategies:
a)Sequential
b)Indexed
c)Linked

<u>Theory</u>

a) Sequential file allocation strategy: In this type of strategy, the files are allocated in a sequential manner such that there is a continuity among the various parts or fragments of the file.

b) Indexed file allocation strategy: In this type of strategy, the files are allocated based on the indexes that are created for each fragment of the file such that each and every similar indexed file is maintained by the primary index thereby providing flow to the file fragments.

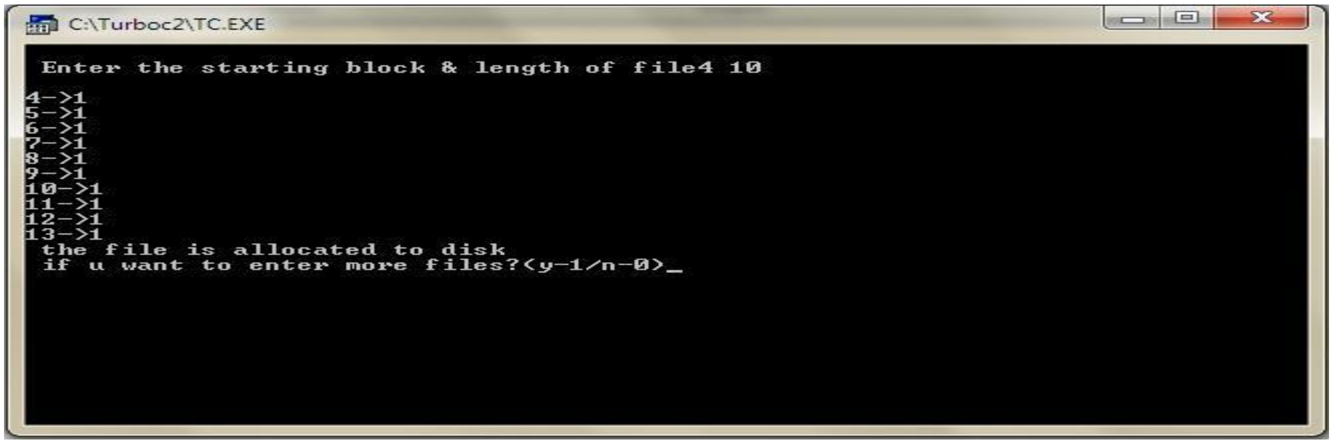
c) Linked file allocation strategy: In this type of strategy, the files are allocated in a linked list format where each and every fragment is linked to the other file through either addresses or pointers. Thus, the starting location of the file serve the purpose of extraction of the entire file because every fragment is linked to each other.

a) Implementation of Sequential File Allocation:

```
#include<stdio.h>
#include<stdlib.h>
main()
{
int f[50],i,st,j,len,c,k;
//clrscr();
for(i=0;i<50;i++)
f[i]=0;
X:
printf("\n Enter the starting block & length of file");
scanf("%d%d",&st,&len);
for(j=st;j<(st+len);j++)
if(f[j]==0)
{
f[j]=1;
printf("\n%d->%d",j,f[j]);
}
else
{
```

```
printf("Block already allocated");
break;
}
if(j==(st+len))
printf("\n the file is allocated to disk");
printf("\n if u want to enter more files?(y-1/n-0)");
scanf("%d",&c);
if(c==1)
goto X;
else
exit(0);
//getch();  }
```

Output:



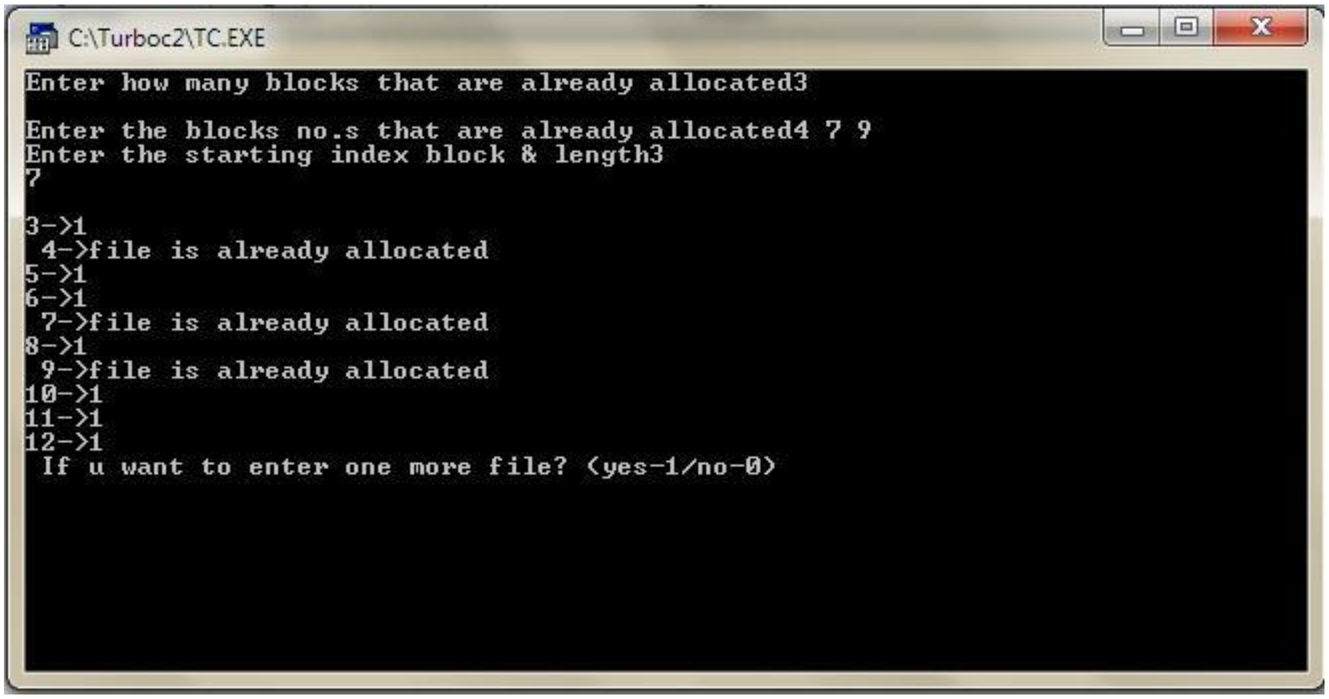
```
 Enter the starting block & length of file4 10
4->1
5->1
6->1
7->1
8->1
9->1
10->1
11->1
12->1
13->1
 the file is allocated to disk
 if u want to enter more files?(y-1/n-0)_
```

b) Implementation of Linked File Allocation:

```
#include<stdio.h>
#include<stdlib.h>
main()
{
int f[50],p,i,j,k,a,st,len,n,c;
//clrscr();
for(i=0;i<50;i++)
f[i]=0;
printf("Enter how many blocks that are already allocated");
scanf("%d",&p);
printf("\nEnter the blocks no.s that are already allocated");
for(i=0;i<p;i++)
{
```

```
scanf("%d",&a);
f[a]=1;
}
X:
printf("Enter the starting index block & length");
scanf("%d%d",&st,&len);
k=len;
for(j=st;j<(k+st);j++)
{
if(f[j]==0)
{
f[j]=1;
printf("\n%d->%d",j,f[j]);
}
else
{
printf("\n %d->file is already allocated",j);
k++;
}
}
printf("\n If u want to enter one more file? (yes-1/no-0)");
scanf("%d",&c);
if(c==1)
goto X;
else
exit(0);
//getch( );
}
```

Output:

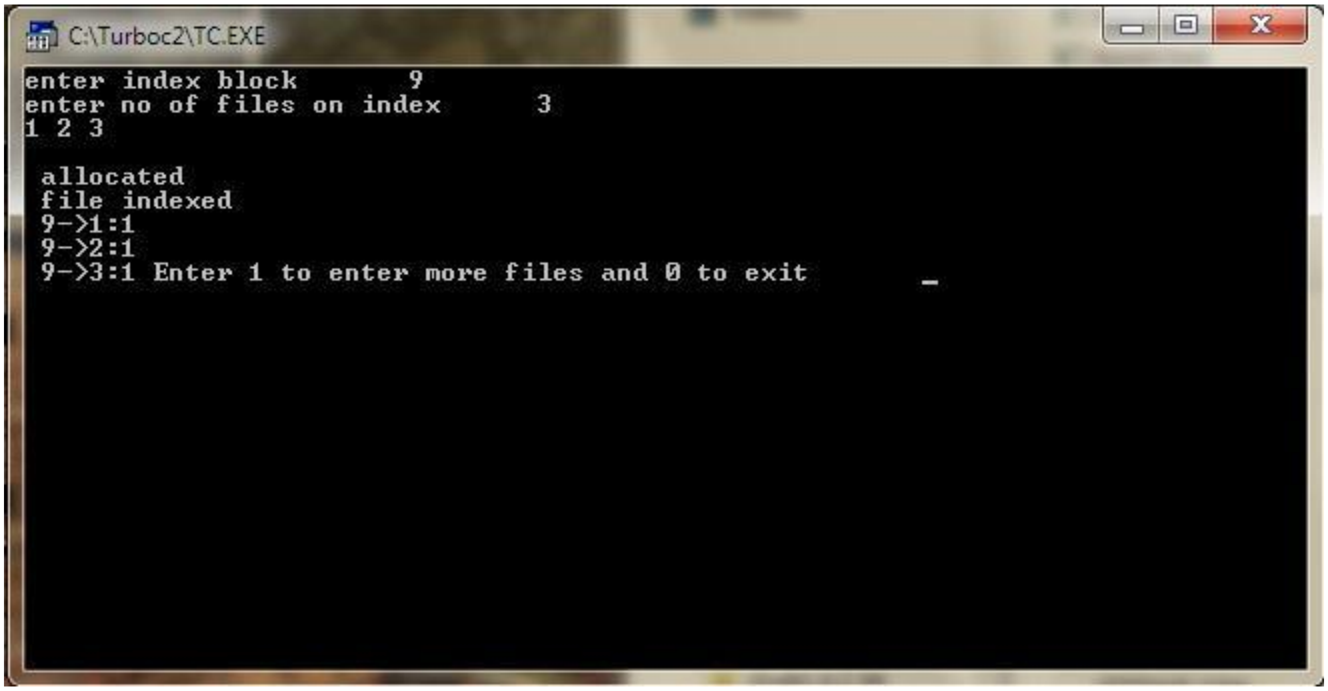


```
Enter how many blocks that are already allocated3

Enter the blocks no.s that are already allocated4 7 9
Enter the starting index block & length3
7

3->1
 4->file is already allocated
5->1
6->1
 7->file is already allocated
8->1
 9->file is already allocated
10->1
11->1
12->1
 If u want to enter one more file? (yes-1/no-0)
```

c) Implementation of Indexed File Allocation:

```
#include<stdio.h>
#include<stdlib.h>
int f[50],i,k,j,inde[50],n,c,count=0,p;
main()
{
//clrscr();
for(i=0;i<50;i++)
f[i]=0;
x:
printf("enter index block\t");
scanf("%d",&p);
if(f[p]==0)
{
f[p]=1;
printf("enter no of files on index\t");
scanf("%d",&n);
}
else
{
printf("Block already allocated\n");
goto x;
}
for(i=0;i<n;i++)
scanf("%d",&inde[i]);
for(i=0;i<n;i++)
if(f[inde[i]]==1)
{
printf("Block already allocated");
goto x;
}
for(j=0;j<n;j++)
f[inde[j]]=1;
printf("\n allocated");
printf("\n file indexed");
for(k=0;k<n;k++)
printf("\n %d->%d:%d",p,inde[k],f[inde[k]]);
printf(" Enter 1 to enter more files and 0 to exit\t");
scanf("%d",&c);
if(c==1)
goto x;
else
exit(0);
//getch();
}
```

Output:



```
enter index block        9
enter no of files on index      3
1 2 3

 allocated
 file indexed
 9->1:1
 9->2:1
 9->3:1 Enter 1 to enter more files and 0 to exit
```