```c
// Prepprocessor

#include<stdio.h>
//#define me 25
int main()
{
   int i;
   //for(i=me; i<me*2; i++)
     // printf("%d",i);

    //printf("%ld",sizeof(i));

    if(sizeof(i)>-1)          //range of an int -32768 to +32767
      printf("hi");
    else
      printf("hello");
    return 0;
}

// Output:         hello
// Explaination:  4> unsigned integer  =>   4 < 0xFFFE    =>   "Hello"
```

```c
// Preprocessor

//.c => Preprocessor =>

#include<stdio.h>
#define me 25
int main()
{
    int i;
    for(i=me; i<me*2; i++)
        printf("%d",i);

     return 0;
}


int main()
{
    int i;
    for(i=25; i<25*2; i++)
        printf("%d",i);
     return 0;
}

```

```
/* FILE (text or binary)*/

// input: io.txt
// output:  out.txt
// a console: con (nick name)



// Case 1. user -> con  => con  (no files are used here) [will not b considered]
// Case 2. user -> con => out.txt
// Case 3. io.txt => out.txt    // file copy
// Case 4. io.txt => con

//Case 4.1:
#include <stdio.h>
int main()
{
   FILE *fp;    // FILE is a system defined data type. Only for files systems.
   char ch;
   fp=fopen("ram.txt","r");     //ram.txt   "hi ayanEOF"
   // "r" reading "w" writing "a" appending

   while(1)
   {
        ch=fgetc(fp);
        if(ch==EOF) break;
        printf("%c",ch);
   }
   fclose(fp);
   return 0;
```

```
}

//Case 4.2:  WAP that will count the no of characters, spaces, new lines, tabs, ',' an so on..
#include <stdio.h>
int main()
{
   FILE *fp;    // FILE is a system defined data type. Only for files systems.
   char ch;
   int nos=0, nota=0, nol=0, noc=0, nod=0;

   fp=fopen("ram.txt","r");     //ram.txt   "hi, .aya.na dey  EOF"
   // "r" reading "w" writing "a" appending

   while(1)
   {
        ch=fgetc(fp);
        if(ch==EOF) break;
        //printf("%c",ch);
        switch(ch)
        {
           case ' ': nos++; break;
           case '\t': notab++; break;
           case '\n': nol++; break;
           case ',': noc++; break;
           case '.': nod++; break;
           default:
        }
   }
   fclose(fp);
```

```
   printf("No of tabs are %d", nos);
   printf("No of tabs are %d", notab);
   printf("No of tabs are %d", nol);
   printf("No of tabs are %d", noc);
   printf("No of tabs are %d", nod);
   return 0;
}

// Case 3. io.txt => out.txt    // file copy
#include <stdio.h>
int main()
{
   FILE *fp;    // FILE is a system defined data type. Only for files systems.
   char ch;
   fp=fopen("io.txt","r");     //io.txt   "hi ayanEOF"
   if(fp==NULL)
   {
      printf("ERROR: SOURCE FILE NOT FOUND\n");
      exit(1);
   }

   ft=fopen("out.txt","w");
   if(ft==NULL)
   {
      printf("ERROR: TARGET FILE NOT FOUND\n");
      fclose(fp);
      exit(1);
   }
```

```
   while(1)
   {
        ch=fgetc(fp);
        if(ch==EOF) break;
        else
         fputc(ch, ft);
   }
   fclose(fp);
   fclose(ft);
   return 0;
}

// Case 2.1. user -> con => out.txt         // "Hi how are you" => out.txt
#include <stdio.h>
#include <string.h>
int main()
{
   FILE *fp;    // FILE is a system defined data type. Only for files systems.
   char s[100];

   fp=fopen("ram.txt","w");
   if(fp==NULL)
   {
      printf("ERROR: FILE NOT FOUND\n");
      exit(1);
   }

   printf("\nPlease Enter some texts\n");
```

```
   while(strlen(gets(s))>0)
   {
        fputs(s,fp);
        fputs("\n",fp);
   }
   fclose(fp);
   return 0;
}

// hi how are you
// r u from kolkata? yes or no ? ...
//



// Case 2.2. user -> con => out.txt          // out.txt => console
#include <stdio.h>
#include <string.h>
int main()
{
   FILE *fp;    // FILE is a system defined data type. Only for files systems.
   char s[100];

   fp=fopen("ram.txt","r");
   if(fp==NULL)
   {
      printf("ERROR: FILE NOT FOUND\n");
      exit(1);
   }
```

```
   while(fgets(s,99,fp))!=NULL)
      puts(s);

   fclose(fp);
   return 0;
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
//Record I/O in Files
// Dealing with Combo Pack of informations

// CODE: Storing information to FILE

#include <stdio.h>
// Making Combo pack of informations
struct emp{
   char name[40];
   int age;
   float bs;
};

int main()
{
   FILE *fp;
   char ch='y';
   struct emp e;       //define a sample of combo pack

   fp=fopen("emp.dat","w");   //dealing with binary files
   if(fp==NULL)
```

```
    {
        puts("\nERROR: CAN'T OPEN FILE\n");
        exit(1);
    }

    while(ch=='y')
    {
        puts("\nEnter your Name");  gets(e.name);
        printf("\nEnter your Age"); scanf("%d", &e.age);
        printf("\nEnter your Age"); scanf("%d", &e.bs);
        fprintf(fp,"%s,%3d,%5.2f",e.name, e.age, e.bs);

        puts("\nAdd another Record (Y/N)\t");          //requesting for another data
        fflush(stdin);
        ch=getche();
    }
    fclose(fp);
    return 0;
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
//Record I/O in Files
// Dealing with Combo Pack of informations

// CODE: Extracting information to FILE

#include <stdio.h>
// Making Combo pack of informations
struct emp{
```

```
    char name[40];
    int age;
    float bs;
};

int main()
{
    FILE *fp;
    char ch='y';
    struct emp e;       //define a sample of combo pack

    fp=fopen("emp.dat","w");   //dealing with binary files
    if(fp==NULL)
    {
        puts("\nERROR: CAN'T OPEN FILE\n");
        exit(1);
    }

    while(fread(&e, sizeof(e),1, fp)==1)
        printf("%s,%3d,%5.2f",e.name, e.age, e.bs);

    fclose(fp);
    return 0;
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// -reczise moves the pointer back by recsize bytes from the current positions
```

```
fseek(fp, -recsize, SEEK_CUR);

// Basically recsize or 0 are just the offsets that tell the compiler
// by how many bytes should the ppinter be moved from a particualr positions
// moves the pointer back by recsize bytes from the current positions

fseek(fp, 0, SEEK_END);
fseek(fp, recsize, SEEK_SET);


position=ftell(fp);

// ftell returns the position as a "long int" which is an offset from the begining of the file

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~`

//Recursion :
// Def: The process in which a function calls itself directly or indirectly is called
// recursion and the corresponding function is called as recursive function.
// E.g.: Sum of natural numbers
// f(n) = 1 + 2 + 3 +……..+ n
// Approach 1:
//    s=s+i;
// Approach 2:
//    f(n) = 1              n=1
//    f(n) = n + f(n-1)     n>1


// C code to implement Fibonacci series
```

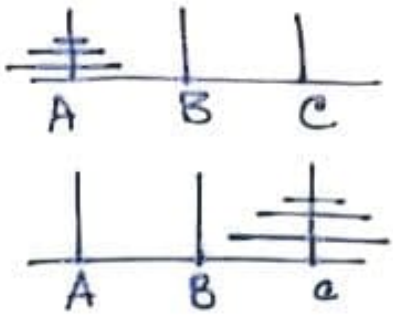
```
#include <stdio.h>

int fib(int n)  // Function for fibonacci
{
        if (n == 0)  // Stop condition
                return 0;

        if (n == 1 || n == 2)  // Stop condition
                return 1;

        else            // Recursion function
           return (fib(n - 1) + fib(n - 2));
}

int main()
{
        int n = 5;
        printf("Fibonacci series of %d numbers is: ",n);

        for (int i = 0; i < n; i++) {
                printf("%d ", fib(i));
        }
        return 0;
}
```

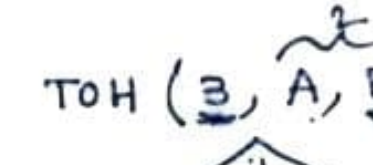
Prob: Towers of Hanoi

```
main()
{
  main();
}
```

Recursion → direct / Indirect

```
main()
{
  sum();
}
sum()
{
  main();
}
```





$f_{10} = f_9 + f_8$

$f_9 = f_8 + f_7$

$f_8 = f_7 + f_6$

:

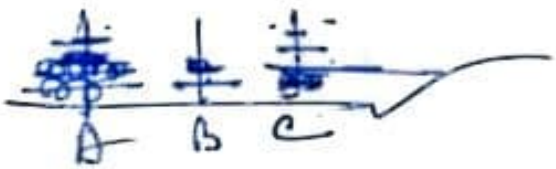
Fibb

0 1 1 2 3 5

$f_2 = f_0 + f_1$

$f_n = f_{n-2} + f_{n-1}$

$f_0 = 0, f_1 = 1$ $n \geq 2$.




```
TOH (n, A, B, C)
{
  if (n == 1)
    print (A → C);
  else
    TOH (n-1, A, C, B);
    print (A → C);
    TOH (n-1, B, A, C);
}
```

when to Stop

when to start