Quick Revision notes UES103 (Sep 2023)

Page 12 for EST notes

HS Pannu

(This is just a basic and minimum material so do not only rely on this if you want to score a high grade)

Computer Memory Hierarchy (CPU registers, cache, RAM, ROM, Hard disk etc)

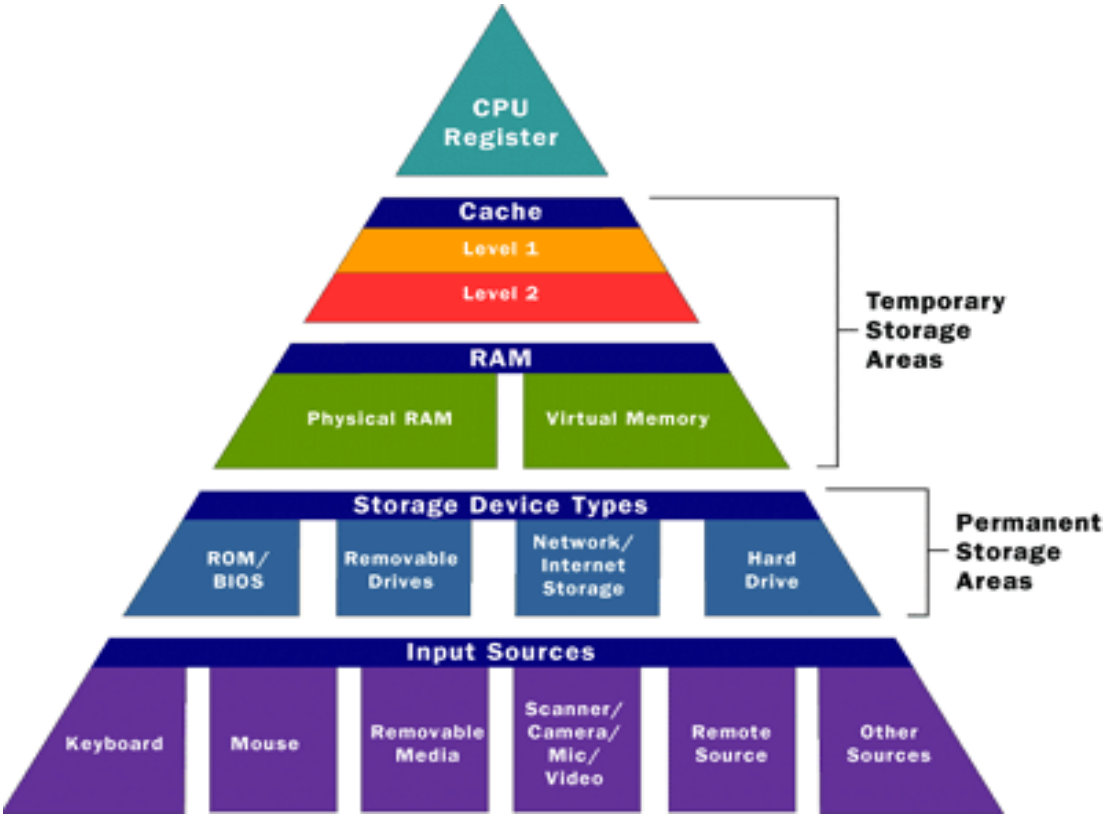

FIGURE 1: Memory hierarchy

Types of Software (System software such as Operating Systems Windows, MAC and Application Software such as MS Office, Compiler etc)

Binary number system

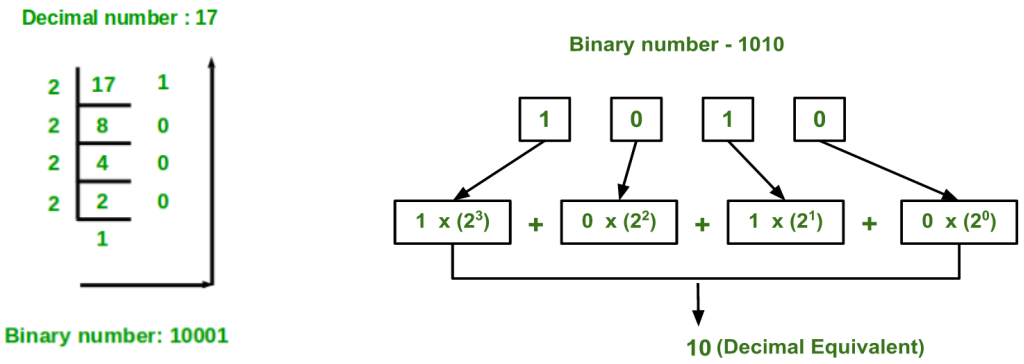

FIGURE 2: Unsigned (only positive) conversions

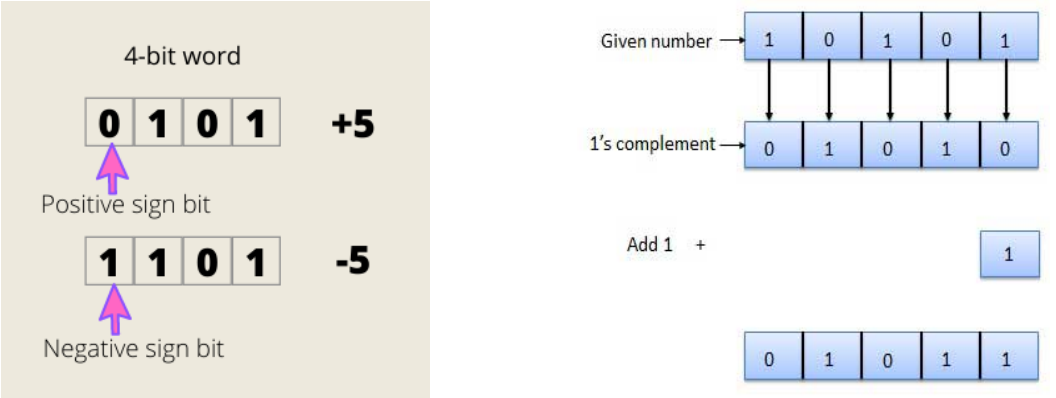

FIGURE 3: Signed conversions: Left is signed bit representation (poor representation) and Right is 2's complement notation.

Algorithm: A set of finite and unambiguous sequence of instructions to perform an action. It may or may not have any input or output (for example?). It may or may not terminate (web based software?). Action performed may be good or bad (computer virus?).

Step 1: Start

Step 2: Create a variable to receive the user's email address

Step 3: Clear the variable in case it's not empty

Step 4: Ask the user for an email address

Step 5: Store the response in the variable

Step 6: Check the stored response to see if it is a valid email address

Step 7: Not valid? Go back to Step 3.

Step 8: End

FIGURE 4: Example algorithm (to do what?)

Flowchart – it is a pictorial view of the algorithm with specific shaped boxes and arrows to indicate the direction of flow.

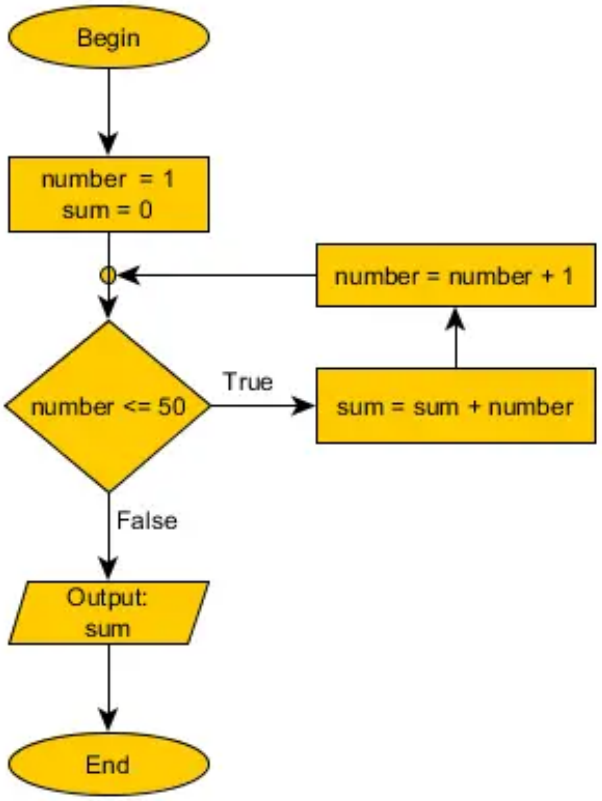

FIGURE 5: Example flowchart (to do what?)

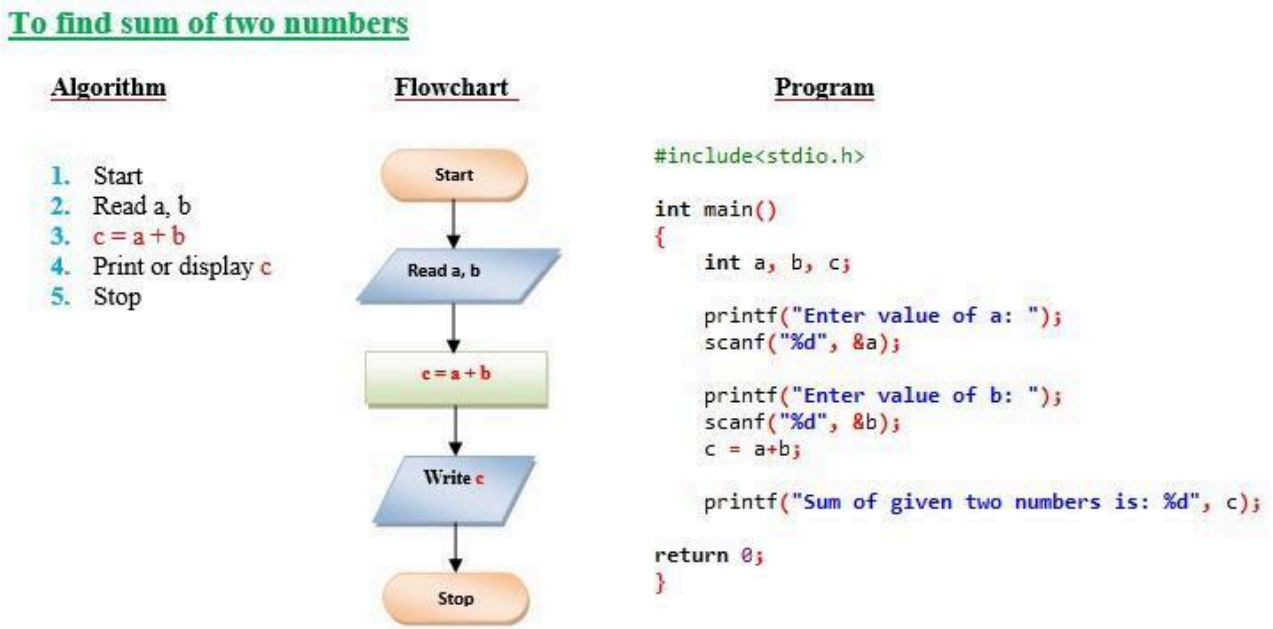

FIGURE 6: Comparison of an algorithm, flow chart and a C program.

Formulate simple algorithms for logical and arithmetic problems – practice the lab assignment programs from https://sites.google.com/a/thapar.edu/uta-007/home/lab-assignments

Structure and Life cycle of a C Program

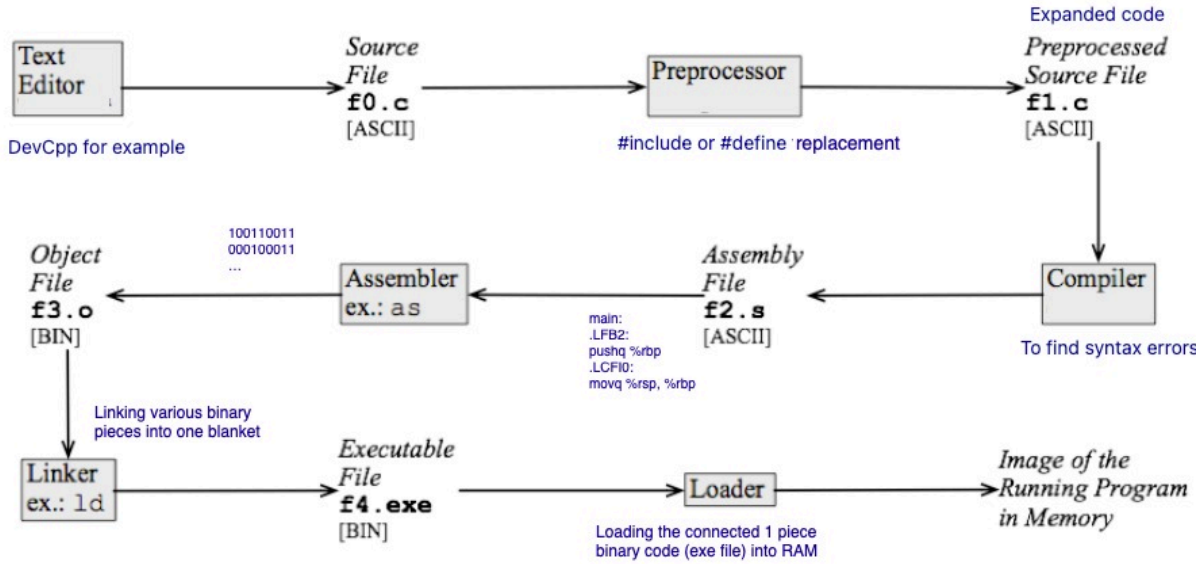

FIGURE 7: Life cycle of a C program

Data types – for example char, int, float, double, long. Do you know their purpose and size in bytes?

Identifiers – is a valid variable name. Id means name and identifier is the one who specifies

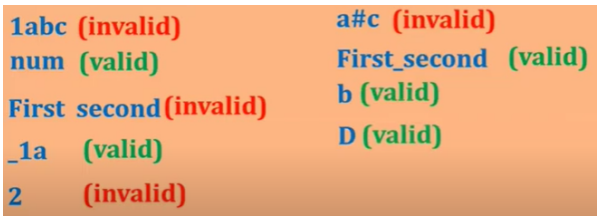

FIGURE 8 – Example of valid and invalid identifiers

Variables – means something whose value can vary. For example int a; float f; char ch; Here a, f, ch are variables.

Keywords – reserved tokens are called keywords such as main, printf, return. They should be used for variable names (why?)

FIGURE 9: Examples of few C language keywords

Constants – whose value never changes for example #define PI 3.14 and then in the program PI will always be 3.14. #define is also a preprocessor

Input/output statements – printf(), scanf(). Do you know %d, %f, %ld, %lf, %c?

Operators

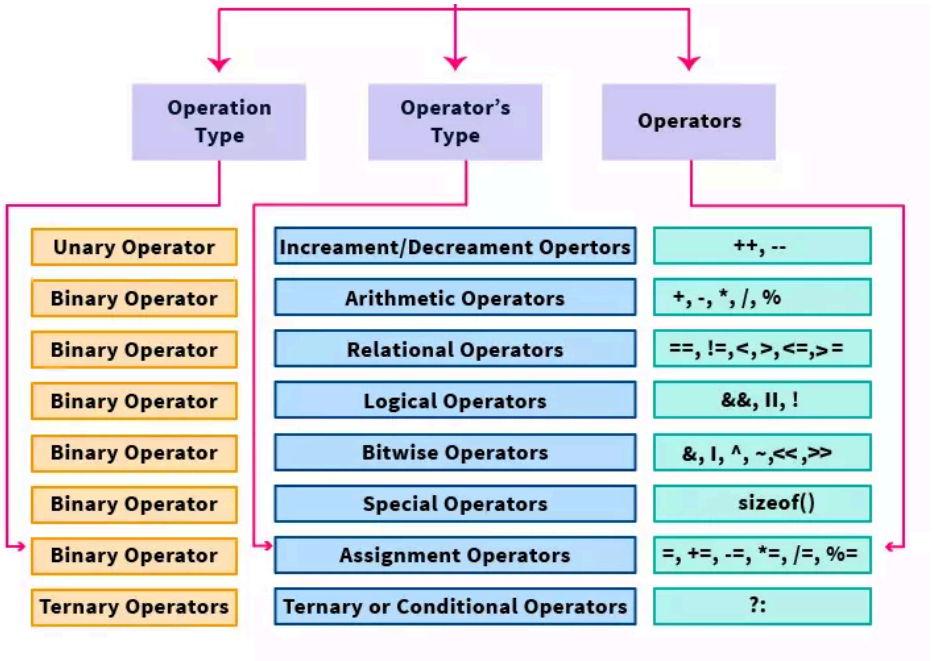

FIGURE 10: Operators classification

Type conversion – is an automatic or implicit conversion done by the compiler internally.

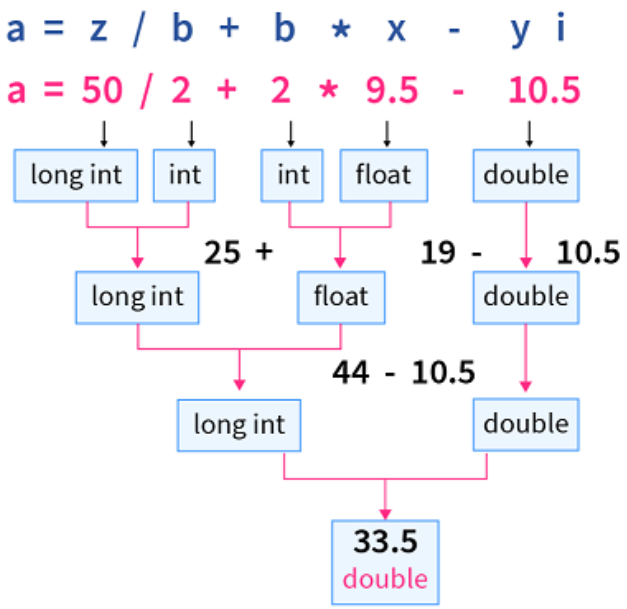

FIGURE 11: Type conversion (automatic)

Type casting – is a data type conversion done by programmer

```
#include<stdio.h>
int main(){
int a=10,b=3;
float f = (float)a/(float)b;
printf("%f",f);        // 3.333333
return 0;
}
```

Translate the algorithms to code snippets - Practice lab assignments

if, if-else - see the program below

```
#include<stdio.h>
int main(){
int a=10,b=-10;
if(a==b) printf("\n Equal");
else printf("\n Different"); // this will get printed
return 0;
}
```

Nested if- else

```
// nested if
#include<stdio.h>
int main(){
int a = 5;
        if(a<100){
                if(a<10) printf("\n %d is less than 10",a);   // nested if. One line so we don't need {}
                else printf("\n %d is smaller than 100 but larger than 10",a);
        }
return 0;
}
```

Multiple if

```
#include<stdio.h>
int main(){
int a = 5;
if(a<100) printf("\n %d is less than 100",a);
if(a<10) printf("\n %d is less than 10",a);
if(a<0) printf("\n %d is a negative integer",a);
return 0;
}
```

else if

```
#include<stdio.h>
int main(){
int a = 150;
if(a<100) printf("\n %d is less than 100",a);
else if(a>200) printf("\n %d is bigger than 200",a);
else printf("\n %d is between 100 and 200",a);
return 0;
}
```

Switch

```
// integer cases
#include<stdio.h>
int main(){
int i=0;
switch(i){
case 1: printf("\t i is 1"); break;
case 0: printf("\t i is 0"); break;
case 10: printf("\t i is 10"); break;
case 2:    printf("\t i is 2"); break;
default: printf("\t i is something else");
}
return 0;
}
```

```
// switch statement with char
#include<stdio.h>
int main(){
char i;
printf("\n Enter a char: "); scanf("%c",&i);
switch(i){
case '#': printf("\t You have entered #"); break;
case 'a':  printf("\t You have entered a"); break;
case 'N':  printf("\t You have entered N"); break;
case '@': printf("\t You have entered @"); break;
case '4': printf("\t You have entered 4"); break;
default:  printf("\t Unknown character"); break;
}
return 0;
}
```

Ternary Operator

```
// ternary operator ?:
// unary means a=a+1 (one variable) or a = -a
// binary a = a+b needs two variables
// string is a word or sentence (array of char) terminated by a null character
#include<stdio.h>
int main(){
int a=10,b=20,c;
c = a<b?(a+10):(b+20);
printf("\n c = %d",c);
printf("\n %s",a>b?"a is smaller than b":"a is bigger than b");
return 0;
}
```

Loops- (while, do-while, for)

```
// there are three types of loops: while, for and do-while
// while loop
#include<stdio.h>
int main(){
int i=0,n=0;
printf("\n How many times do you want to see the greetings? ");
scanf("%d",&n);
```

```
 while(i<n){
  printf("\n Good evening");
  i++;
 }
return 0;
}
```

// while loop

```
#include<stdio.h>
int main(){
int n,i=0;
printf("\n How many times do you need the greeting? ");
scanf("%d",&n);
while(i<n){
 printf("\n Good evening");
 i++;
}
return 0;
}
```

// do-while program. If n=0 even then it will print the message once

```
#include<stdio.h>
int main(){
int i=0,n=0;
printf("\n How many times do you want to see the greetings? ");
scanf("%d",&n);
 do{
  printf("\n Good evening");
  i++;
 }while(i<n);

return 0;
}
```

// for loop is similar to while loop (condition first and then do-part)

```
#include<stdio.h>
int main(){
int i=0,n=0;
printf("\n How many times do you want to see the greetings? ");
scanf("%d",&n);
 for(i=0;i<n;i++){
  printf("\n Good evening");
 }
return 0;
}
```

Nesting of Loops

```
// nested loops: print an array three times
#include<stdio.h>
int main(){
int i,j,n=3;
int A[] = {11,22,33};
for(i=0;i<n;i++){
        for(j=0;j<n;j++){
```

```
                printf("\t%d",A[i]);
        }
        printf("\n"); // new line after each row
}
return 0;
}
```

break, continue

```
// break statement is used for loops and switch block
#include<stdio.h>
int main(){
int n;
while(1){  // infinite loop
        printf("\n Enter a non zero integer: ");
        scanf("%d",&n);
        printf("\n You have entered:  %d",n);
        if(n==0) break;  // break out of the loop if n is 0
}
return 0;
}
```

```
// continue is to finish current iteration and start the next one
#include<stdio.h>
int main(){
int i;
for(i=0;i<10;i++){
        if(i%3==0) continue; // continue to next iteration if i is div by 3
        else printf("\t %d",i);
}
return 0;
}
// 1 2 4 5 7 8 will be the output
```

goto – this should not be used as it results in jumping from anywhere to anywhere and code becomes confusing, also known as spaghetti code (like noodles)

```
#include<stdio.h>
int main(){
printf("\n Start");
Delhi: printf("\n in Delhi");
goto Goa;
printf("\n Pune"); // Pune will be skipped
Goa: printf("\n at Goa");
return 0;
}
```

Implement the switch () to solve the basic functions of scientific calculator.

```
// Calculator using switch statement
#include<stdio.h>
int main(){
```

```
int a,b;
char ch;
printf("\nEnter the operator (+,-,*,/,%): "); scanf("%c",&ch);
printf("\nEnter two numbers: "); scanf("%d%d",&a,&b);
switch(ch){
case '+': printf("\t ANS: %d",a+b); break;
case '-': printf("\t ANS: %d",a-b); break;
case '/': printf("\t ANS: %f",(float)a/(float)b); break;
case '*': printf("\t ANS: %d",a*b); break;
case '%': printf("\t ANS: %d",a%b); break;
default: printf("\t invalid operator");
}
return 0;
}
```

One-dimensional array

```
// arrays - it is a collection of similar data type such as int, char, float etc.
// it should be homogeneous (we cannot mix different data types)
// All the elements are adjecent such as A[0], A[1],....A[n-1] for n elt array
#include<stdio.h>
int main(){
int A[5],i,n=5; // A[5] declaration of array of 5 ints
printf("\n This program is to demonstrate int array \n");

for(i=0;i<n;i=i+1){  // i = 0,1,2,3,4 ....three types of loops?
 printf("\n Enter an integer: "); scanf("%d",&A[i]); // A[0], A[1],...
}
printf("\nNow printing the array of 5 integers: ");
for(i=0;i<n;i++)
 printf(" %d ",A[i]);

return 0;
}
```

// Array initialization

```
#include<stdio.h>
int main(){
int A[] = {22,44,-8,12,6};
int i,n=5;

printf("\nNow printing the array of 5 integers: ");
for(i=0;i<n;i++)
 printf(" %d ",A[i]);

return 0;
}
```

Input and output of a string

```
// gets and puts functions
#include<stdio.h>
```

```
int main(){
char S[50];
puts("Enter a string including blank spaces: ");
gets(S);
puts("You have entered: ");
puts(S);
return 0;
}
```

```
// intializing a string
#include<stdio.h>
int main(){
char S[] = "Thapar Institute Patiala 147004";
puts("The given string is: ");
puts(S);
return 0;
}
```

```
// printing charcter by character of string
#include<stdio.h>
int main(){
char S[] = "Thapar Institute Patiala 147004";
int i,n = strlen(S);
for(i=0;i<n;i++) printf("%c",S[i]);
return 0;
}
```

string inbuilt functions

```
// strings - inbuilt functions
#include<stdio.h>
#include<string.h>
int main(){
char S[] = "hello 123", T[] = "abc 000", Z[100];
printf("\n\t STRLEN of S %d",strlen(S)); // length of S
printf("\n\t STRCMP %d",strcmp(T,S)); // if S eq. T then 0, otherwise -1 or 1
printf("\n\t STRCAT %s", strcat(S,T)); // concatinate means merge
printf("\n\t STRREV = %s", strrev(T));  // it reverses string T
printf("\n\t STRCPY = %s",strcpy(Z,T)); // copy T into Z
return 0;
}
```

// ASCII values - American Standard

// ASCII (American Standard Code for Information Interchange) is the most common character encoding format for text data in computers and on the internet.
// ASCII - it is a character to integer mapping
// American Standard Code for Information Interchange
// There are total 256 characters including printable and control characters
// There for 8 bits or 1 byte can define them (which is size of a char variable)

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

```
int i;
for(i=0;i<256;i++)
 printf("\n %d ASCII belongs to char = %c", i,i);

return 0;
}
```

—---------- EST —----------

Syllabus (from course website https://sites.google.com/a/thapar.edu/uta-007

Functions: Function prototype, Definition and Call, Type of Functions, Scope of variables in (Block, Function, Program, File), Storage classes (Auto, Register, Static and Extern), Recursion (with the introduction of Stack), Implementation of recursion to solve the problem of Tower of Hanoi.

Two-dimensional and its operations (Addition, Transpose and Multiplication), Passing of array into a function (row and entire array), 2-D Character array.

Pointers: Introduction to Pointers, Pointer arithmetic, Passing arguments to a function using pointer (understanding of call by value and call by reference), Accessing arrays using pointers Dynamic memory allocation (malloc(), calloc(), realloc() and free()), Pointer and Functions.

Structures and Union: Structure declaration, Initialization of structures, Structure variables, Accessing structure elements using (.) operator, Array of structure variables, Passing structure variable to a function (individual and entire structure), Structure pointer, Comparison of Structure and Union.

File Handling: Introduction of Files (streams in C), using File (Declaring, Opening and Closing), Operations on File (Reading, Writing and appending), and Random Access of a file, command line argument.