**POINTERS**

**1. Array of pointers to store string.**

e.g.

#include <stdio.h>

const int MAX = 4;

int main () {

char \*names[] = {

"Zara Ali",

"Hina Ali",

"Nuha Ali",

"Sara Ali"

};

int i = 0;

for ( i = 0; i < MAX; i++) {

printf("Value of names[%d] = %s\n", i, names[i] );

}

return 0;

}

**2. Return Pointer from function.**

It is not a good idea to return the **address of a local variable** outside the function, so you would have to define the local variable as **static** variable.

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

/\* function to generate and return random numbers. \*/

int \* getRandom( ) {

static int r[10]; // this variable is declared as static as we are returing its //address outside the function

int i;

/\* set the seed \*/

// srand( (unsigned)time( NULL ) );

for ( i = 0; i < 10; ++i) {

r[i] = rand();

printf("%d\n", r[i] );

}

return r;

}

/\* main function to call above defined function \*/

int main () {

/\* a pointer to an int \*/

int \*p;

int i;

p = getRandom();

for ( i = 0; i < 10; i++ ) {

printf("\*(p + [%d]) : %d\n", i, \*(p + i) );

}

return 0;

}

**Dynamic memory allocation:**

**1. malloc :** stands for memory allocation.

The function **malloc()** reserves a block of memory of specified size and return a pointer of type **void**, which can be casted into a pointer of any form.

syntex: ptr =(cast\_type \*) malloc (size);

mallocl() does not initialize the memory location with any value. i.e. the memory allocate with the hepl of malloc() contains garbage value.