**DYNAMIC MEMORY ALLOCATION IN C**

The concept of dynamic memory allocation enables the c programmer to allocate memory at run time. Dynamic memory allocation in c language is possible by 4 functions of stdlib.h header files.

1. Malloc()
2. Calloc()
3. Realloc()
4. Free()

Before learning this lets see what is the diff b/w static memory allocation & dynamic memory allocation…..

Static memory

1. allocated compile time
2. memory cant be increased while executing program
3. means store the fixed memory status here in static memory.

Dynamic

1. memory is allocated at run time
2. memory can be increase while execution (run)program

now berfly describe malloc()in c

1. malloc()

dee gaye size of bytes se memory me 1 block create ya allocate hoga..Dynamically

malloc() y ek pointer return karta he jo user k according jis bhi data k type ko use karna chahte he memory me dynamically 1st byte allocate krwade run time par.

Syntax void \* malloc(size in bytes);

Jo bhi size diya he uske heesab se 1 block return karega void means void pointer kisi bhi D.T me pointer ko allocate kara sakta h.

Jesa ki hum jante he malloc() void pointer return karta he jisko hume cast type ki help es convert karna h jis bhi taarah ka data type ho..

Example… With using datatype in malloc()

Ptr=(cast type \*)malloc(size in bytes);

Cast typemeans jis bhi type ka data type use karwana he.

Size in bytes means 1 block allocate karega jo datatype k byte khisaab se allocate karega.

Jese humne malloc se size banaya

Ptr=(int \*)malloc(10);

Ten n y garbej value dega because

Cast type k accoding size 2 byte hoti h then 10\*2=20..& 5\*2=10

Wap using malloc()

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

Int main()

{

Int \*p;

P=(int \*)malloc(sizeof(int));

If(p==NULL)

{

Printf(“fail to allocated memory”);

}

Else

{

\*p=20;

Printf(“%d”,\*p);

// we can use free (pointer parameter is here) for free particular pointer in this parameters.

Getch();

Return 0;

}

}

**here 20 output with storing dynamic type**

**malloc() ka use kare ya kisi or ka pr jab hum memory allocate karte he to y pointre return katra he. allocate ki gai memeory k 1 st byte ko point karta he.**

**to isliye in sare functons ka return type ptr hota h…..**

**using char with malloc()**

3 Header files

Int main()

{

Char \*p;

Int length;

Puts(“enter the l;ength of string”);

Scanf(“%d”,&length);

P=(char\*)malloc(sizeof(char)\*length+1);

If(p==NULL)

{

Puts(“memory is not available”);

}

Else

{

Fflush(stdin); // if any character present in the buffer then it will be cleared that purpose we using fflush() function here..

Puts(“enter the string”);

Gets(P);

Puts(p);

}

And we can free () also this segment

Using free(p);

Getch();

Return 0;

}

**Calloc ()**

The name calloc() stands for “contiguous allocation”.

The diff between malloc and calloc is that mallloc() allocates single block of memory or calloc() allocates multiples off blocks of memory each of same size and sets all bytes…

Syntax ptr=(cast type\*)callac(n,elements size);

Like ptr=(float\*)calloc(25,sizeof(float));

This stmt allocates contiguous space in memory for an 25 elements each odf size of float i.e. 4 bytes….

**Wap to define calloc() in c**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

/\* void \* calloc( no. of items,sizeof each items);

Void free(current storage pointer); /\*

Int main()

{

Int \*marks,length,counter;

Puts(“enter the number of subjects”);

Scanf(“%d”,&length);

Marks=(int \*)calloc(length,sizeof(int));

If(marks==NULL)

{

Puts(“unable to allocate memory”)

}

Else

{

For(counter=0;counter<length;counter++)

{

Printf(“enter the marks of %d sub”,counter+1);

Scanf(“%d”,marks[counter]);

}

For(counter=0;counter<length;counter++)

{

Printf(“%d\n”,marks[counter]);

}

}

// Free(marks);

Getch();

Return 0;

}

Output will be like arrays element

**Realloc ()**

Sizeof dynamic memory allocation can be changed by using realloc()

Realloc() reallocates the old object pointed to by malloc() functcion and returns a particular size of specified new same pointer..

Have a look for using program

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

Int main()

{

Char \*x;

X=(char \*)malloc(13\*sizeof(char));

If(x==NULL)

{

Printf(“unable to allocate memory”);

}

Else

{

Strcpy(x,”learning lad”);

Puts(x);

X=(char\*)realloc(x,19);

Strcat(x,”rocks”);

Puts(x);

Free(x);

}

Return 0;

}

Then we will be output particular increasing string length ……..rocks next time..

**Free()**

**Dynamically allocated memory created with either calloc() or malloc() doesent get freed on its own.**

**We muist explicitly use free() to release the space…**

**Syntax free( pointer name as a parameters);**