Dynamic Memory Allocation

↓

?

Run Time Memory Alloc

Memory Alloc

static

int a;

int arr[10];

dynamic

(done with pointer)

How C Helps In DMA ?

pre-defined fn: malloc ( )

header file: alloc.h, stdlib.h, malloc.h

alloc.h → In TC

stdlib.h, malloc.h → In MinGW

prototype/decl : void* malloc(size_t);

Ret Type name argument type

① What is size_t ?

Before answering this, let us answer the following:

② What is typedef ?

typedef is a keyword which allows a programmer to allot alternate name to data types.

Syntax:

```
typedef <old name> <new name>;
```

Ex:

```
typedef int number;
typedef float decimal;
```

```
typedef int number;

number main( )
{
    number a = 10, b = 20;

    ≡

    ret 0;
}
```

```
typedef   unsigned int   size_t;
```

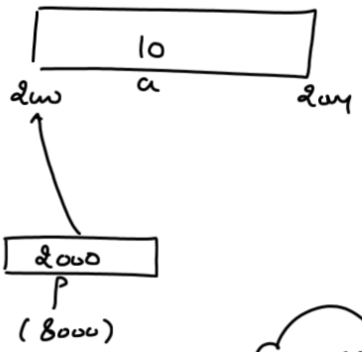
1. string.h
2. stdio.h
3. alloc.h
4. stdlib.h
5. malloc.h

What is a void * ?

```
int a;
char b;
float c;
int *p;
p = &a;
char *q;
p = &b;
float *r;
r = &c;
```

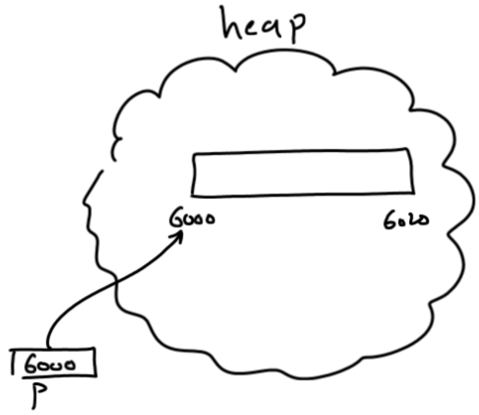
```
int a;
char b;
float c;
void * p;
p = &a; ✓
:
p = &b; ✓
:
p = &c; ✓
```

```
  int  a=10; ✓
  void  *p; ✓
  p =  &a; ✓
X printf ("%d", *p);
✓ printf("%d", *((int *)p));
```

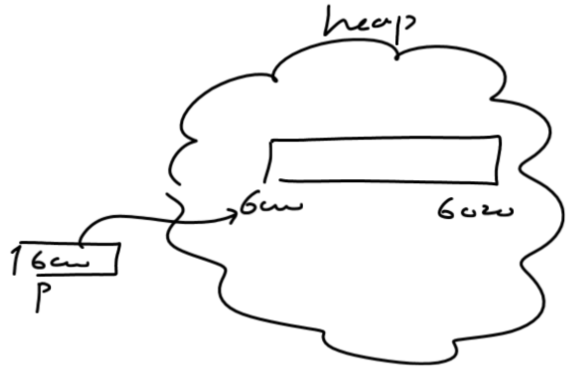


We cannot de-reference a void * easily!

```
  int  main( )
  {
    void  *p;
✓ p=  malloc (20);
X *p= 10;
X p++;
```



```
int main()
{
   int *p;
✓ p=(int*)malloc(20);

✓ *p=10;
✓ p++;
```



WAP to create a dynamic array of 'n' integers where 'n' should be taken from the user. Now ask the user to input values in that array and finally display all array values along with their sum and average.

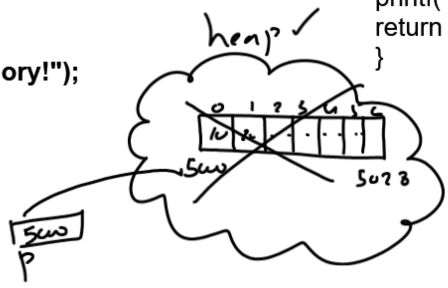
```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   int *p,n,i,sum=0;
   printf("How many numbers ?");
   scanf("%d",&n);
   p=(int *)malloc(n*sizeof(int));
   if(p==NULL)
   {
      printf("Insufficient memory!");
      return 1;
   }

   for(i=0;i<n;i++)
   {
      printf("Enter no:");
      scanf("%d",p+i);
   }
```

NULL —> 0

0

```
for(i=0;i<n;i++)
   {
      printf("\n%d",*(p+i));
      sum=sum+*(p+i);
   }
free(p);
printf("Sum is %d",sum);
printf("\nAvg is %f",(float)sum/n);
return 0;
}
```

? —> free(p);



What is dangling pointer ?

```
int *p;

*p=10;
```

Wild ptr

```
int *p;

p=(int *) malloc (20);
.
.
.
free(p);
*p=10; ?
```

This p now becomes a dangling point!

Unpredictable!