C_80 ⇒ Void Pointers in C

int *ip; ⇒ pointer to int
float *fp; ⇒ pointer to float
char *cp; ⇒ pointer to char

⇓

* These pointers have associated datatype within it.

* But void pointers don't have associated datatype.

Void Pointer Declaration :-

```
Void *vp;
```

* void pointer can store address of anyother datatype; it is not storing specific datatype.

```
int a; float b;
int *p = &b;   ✗
int *P = &a;   ✓
```

↓

Normal Pointer

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

↓

void pointer

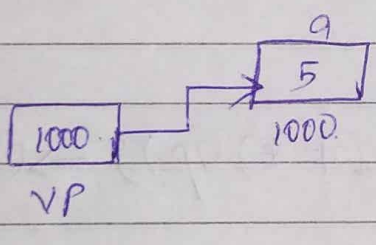
* It is like a generic pointer (i.e) we can typecast this (i.e) in empty glass we can store water or milk or juice.

* But we cannot de-reference a void pointer; it will give error.

Eg: int a;
void *vp;
vp = &a;
printf("%d", *vp); ⇒ Error ×
(We cannot dereference void pointer.)

* So, first we have to typecast or convert.

Eg: ① int a;
void *vp;
vp = &a;
printf("%d", *(int *)vp); ⇒ (5).



Example ②:

float b = 1.11;
void *vp;
vp = &b;
printf("%f", *(float *)vp); ⇒ (1.11).

Example ③:

char c = 'A';
void *vp;
vp = &c;
printf("%c", *(char *)vp); ⇒ 'A'

Use of void pointer;

* In a single program, if we use int, char, float; for all 3 we declare 3 different pointers.

* But using void pointer; we can store all the datatypes in a single void pointer.

Note:
By using one pointer we access 3 datatypes

Example:

```
int a=5;
float b=1.11;
char c='A';

void *vp;
vp=&a;
printf("%d",*(int*)vp); => 5

vp=&b;
printf("%f",*(float*)vp); => 1.11

vp=&c;
printf("%c",*(char*)vp); => A
```

Notes:

* By using a single void pointer we can access multiple datatypes by doing typecasting process.

* Malloc & Calloc are built in functions for dynamic memory allocation; malloc & Calloc returns void pointers to the memory which are discussed in dynamic memory allocation.

PROBLEM 1:

```c
#include <stdio.h>
#include <stdlib.h>
/** VOID POINTERS **/
int main()
{
    int a=5;
    float b=1.11;
    char c='A';
    void *vp;

    vp=&a;
    printf("Value at address of vp:%d\n",*(int *)vp);

    vp=&b;
    printf("Value at address of vp:%f\n",*(float *)vp);

    vp=&c;
    printf("Value at address of vp:%c\n",*(char *)vp);

    getch();
}
```

```
"D:\1. C NOTEBOOK\C LANGUAGE\C PROGRAMS\PART 5_Jennys Lectures\PART 6_JENNYS LECTURE_POINTERS\9_VOID POINTER
Value at address of vp:5
Value at address of vp:1.110000
Value at address of vp:A
```