Sunbeam Institute of Information Technology
Pune and Karad

Module - Embedded C Programming

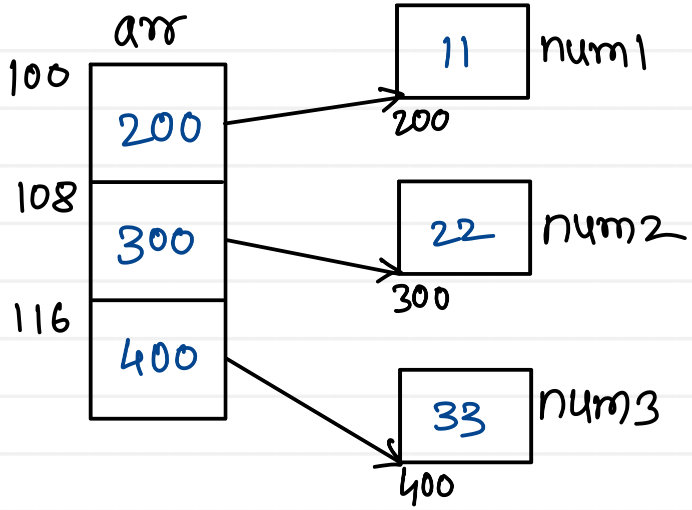
Trainer - Devendra Dhande

Email – devendra.dhande@sunbeaminfo.com

Array of pointers

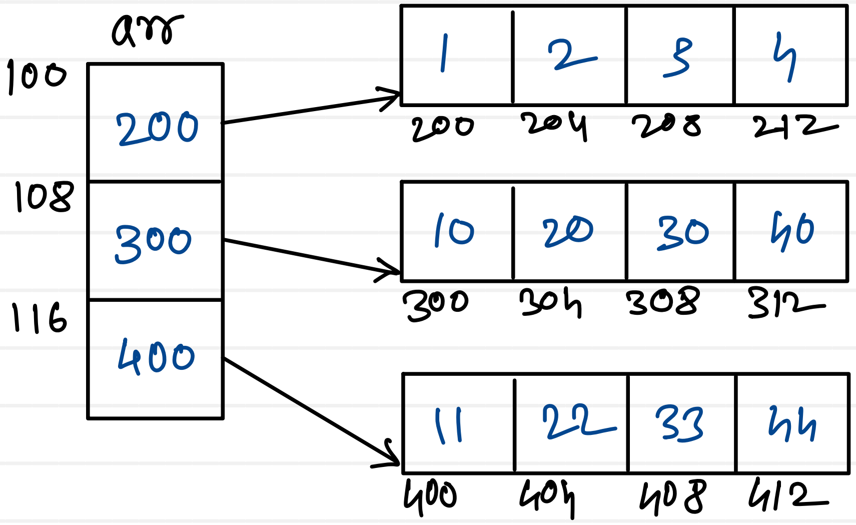
<data type> <name>[length];

int *arr[3]



arr = 100
arr[0] = 200
arr[1] = 300
arr[2] = 400

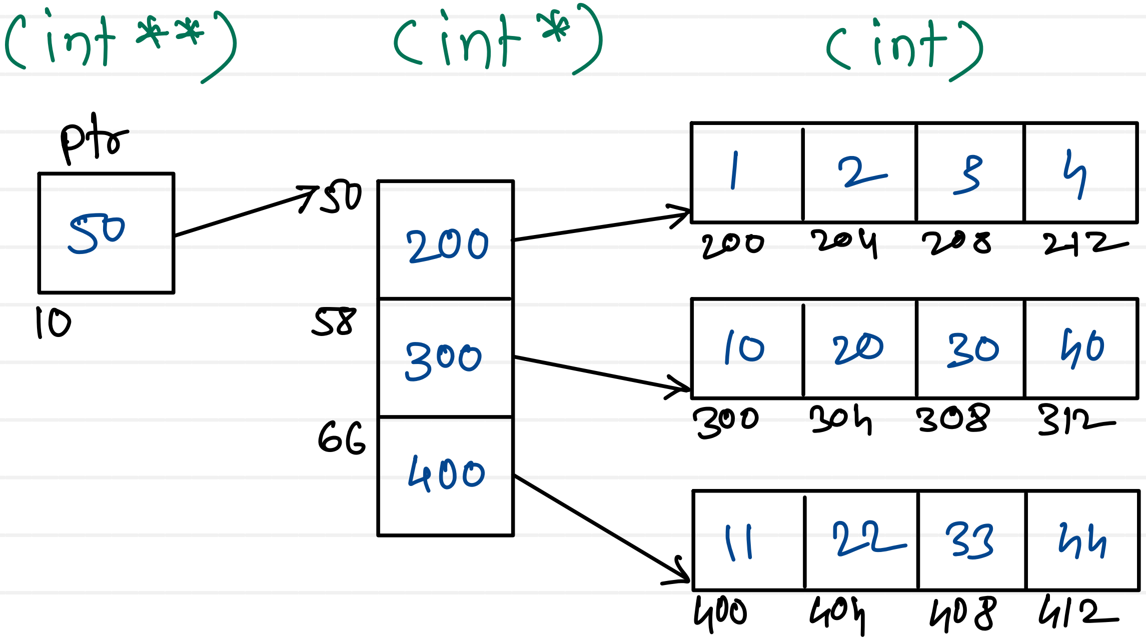
*arr[0] = 11
*arr[1] = 22
*arr[2] = 33



arr = 100
arr[0] = 200
arr[1] = 300
arr[2] = 400

i = 0, 1, 2, 3
arr[0][i] = 1, 2, 3, 4
arr[1][i] = 10, 20, 30, 40
arr[2][i] = 11, 22, 33, 44

Dynamic memory allocation for 2D array



ptr=50   *ptr=200   => ptr[0]

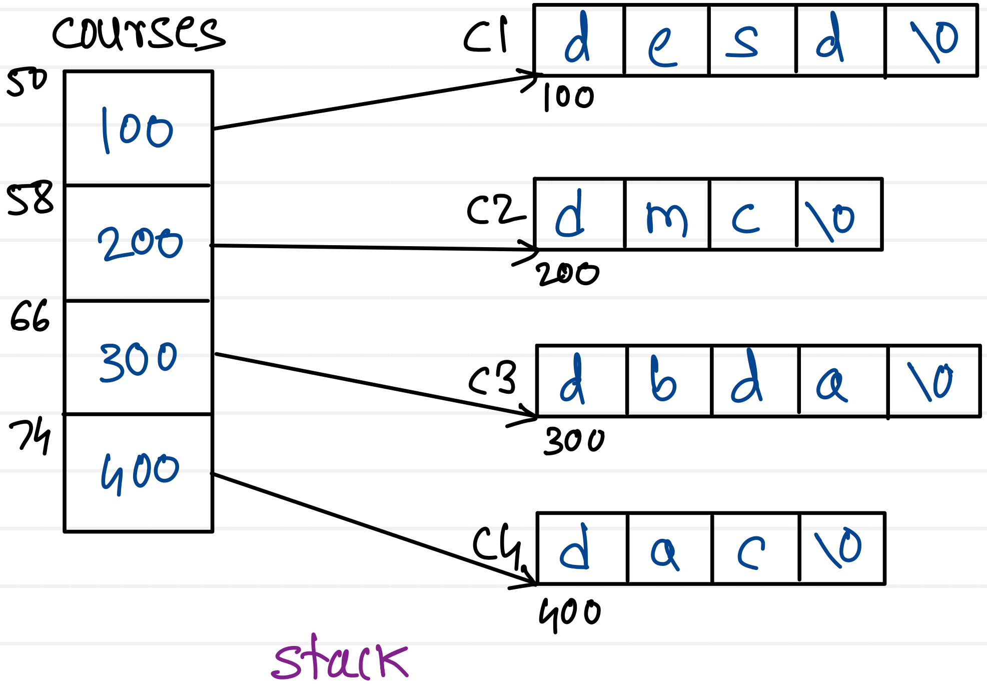
ptr+1=58   *(ptr+1)=300   => ptr[1]

ptr+2=66   *(ptr+2)=400   => ptr[2]

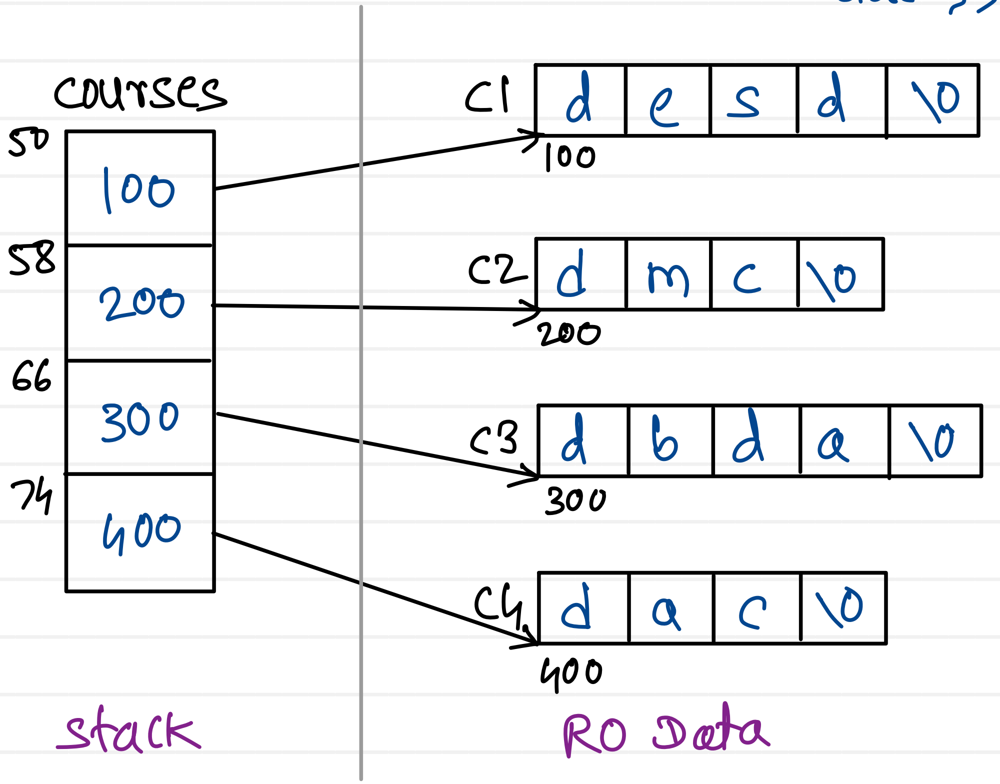
```
int **ptr = NULL;
ptr = (int **)malloc( 3 * sizeof(int *));
                              3*8 = 24 bytes
for (i=0; i<3; i++)
    ptr[i] = (int *)malloc( 4*sizeof(int));
                              4*4 = 16 bytes

for (i=0; i<3; i++)
    free(ptr[i]);
free(ptr);
ptr = NULL;
```

Array of pointers

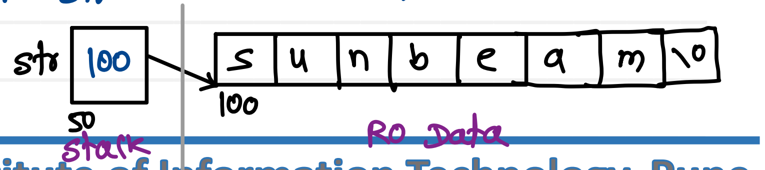
```
char c1[ ] = "desd";
char c2[ ] = "dmc";
char c3[] = "dbda";
char c4[] = "dac";

char *courses[ ] = { c1, c2, c3, c4 };
```



```
char *courses[ ] = { "desd", "dmc", "dbda",
                                      "dac" };
```



```
char *str = "sunbeam";
```



Command line arguments

- command line arguments are passed to the main function.

- int main(void)
- int main(int argc, char *argv[])

  argc - count of command line args.

  argv - list/array of cmd line args

- int main(int argc, char *argv[], char *envp[])

  envp - list/array of environment variables

- main is an entry point function where your program starts executing.

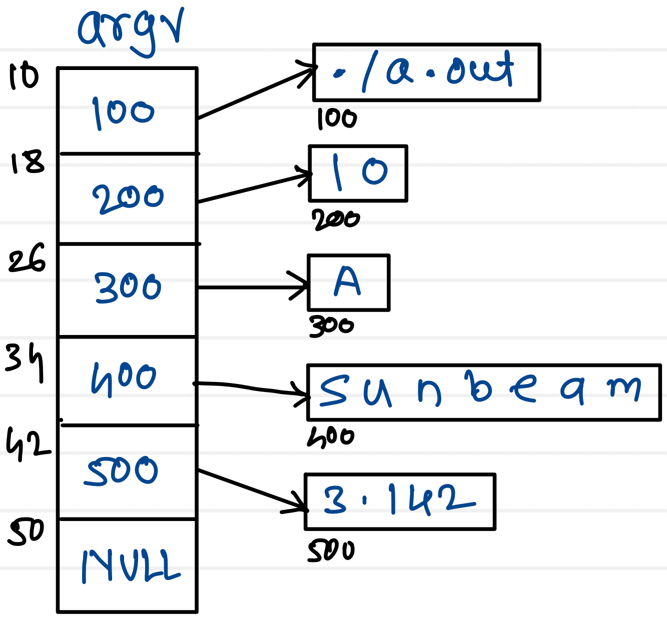
- main is called as callback function because, we declare & define this function into code but never called into our code.

Run program as :

```
./a.out 10 A sunbeam 3.142
```

- first command line argument is always name of the program.

argc = 5



```
int *ptr = (int *)malloc(20);
        ↳ array of 5 integers

char *ptr = (char *)malloc(20);
        ↳ array of 20 characters

void *ptr = malloc(24)
        (int *)ptr → array of 6 interger
        (double *)ptr → array of 3 double vars

void **ptr = malloc(24)
        ↳ array of 3 void pointers

int **ptr = malloc(24)
        ↳ array of 3 int pointers

int **ptr = malloc(20)   — on 64 bit, unexpected
                           -on 32bit - array of 5 pointers
```

Structure

- structure is a user defined data type.
- structure is collection of similar or dis-similary type of data which is logically related in contiguous space.
- struct keyword is used to create a type
- syntax :

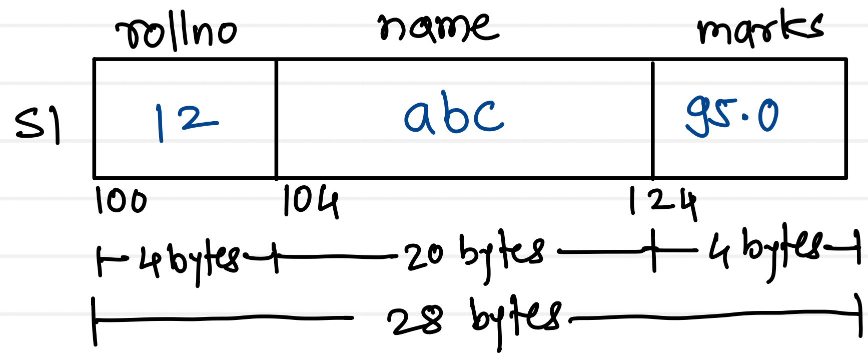
```
struct <name> {
            memb1 ;
            memb2;
            :
};
```

- structure members are accessed with '.' or '->' operator.

```
struct student {
        int rollno ;
        char name[20];
        float marks ;
};
```

type declaration

struct student s1 = { 12, "abc", 95.0f };



- size of structure variable is sum of sizes of its members.

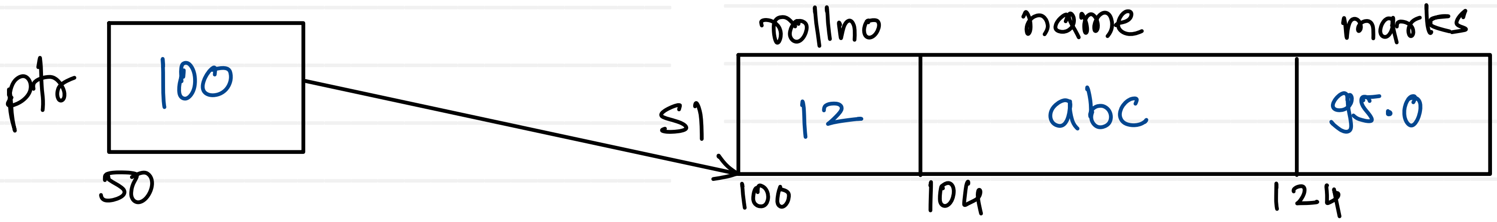
Pointer to structure

'.' operator is used to access members through variable

'->' operator is used to access members through pointer

```
struct student {
        int rollno;
        char name[20];
        float marks;
};
```

```
struct student *ptr = &s1;        struct student s1 = {12, "abc", 95.0f};
```



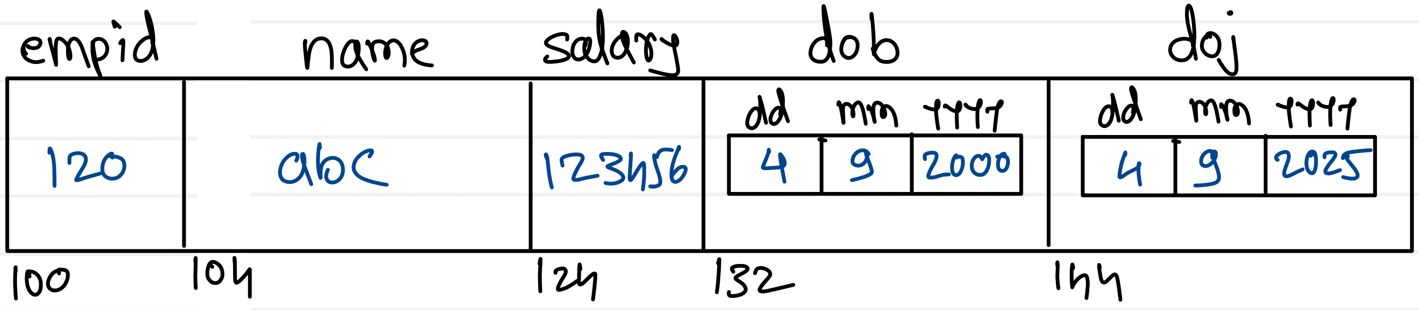
```
ptr -> rollno = 12            s1.rollno = 12
ptr -> name = "abc"           s1.name = "abc"
ptr -> marks = 95.0           s1.marks = 95.0
```

Nested structure

```
struct date {
        int dd;
        int mm;
        int yyyy;
};

struct employee {
        int empid;
        char name[20];
        double salary;
        struct date dob, doj;
};

emp.dob = {4, 9, 2000}
emp.doj = {4, 9, 2025}

emp.dob.dd = 4          emp.doj.dd = 4
emp.dob.mm = 9          emp.doj.mm = 9
emp.dob.yyyy = 2000     emp.doj.yyyy = 2025
```

```
struct employee {
        int empid;
        char name[20];
        double salary;
        struct date {
                int dd;
                int mm;
                int yyyy;
        } dob, doj;
};
```

```
struct employee emp;
```



empid	name	salary	dob	doj
120	abc	123456	dd mm yyyy 4 9 2000	dd mm yyyy 4 9 2025
100	104	124	132	144

SUNBEAM

Thank you!!!

Devendra Dhande

devendra.dhande@sunbeaminfo.com