Programming for Problem Solving (PPS)
GTU # 3110003

USING

{C}
Programming

Array & Strings

Prof. Nilesh Gambhava

Computer Engineering Department,
Darshan Institute of Engineering & Technology, Rajkot

Need of Array Variable

- Suppose we need to store `rollno` of the student in the integer variable.

Declaration
```
int rollno;
```

- Now we need to store `rollno` of 100 students.

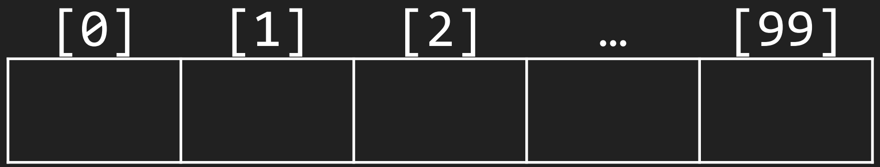
Declaration
```
int rollno101, rollno102, rollno103, rollno104...;
```

- This is not appropriate to declare these many integer variables.
  e.g. 100 integer variables for `rollno`.
- Solution to declare and store multiple variables of similar type is an array.
- An array is a variable that can store multiple values.

Definition: Array

- An array is a fixed size sequential collection of elements of same data type grouped under single variable name.

```c
int rollno[100];
```



Fixed Size	Sequential	Same Data type	Single Name
Here, the size of an array is 100 (fixed) to store rollno	It is indexed to 0 to 99 in sequence	All the elements (0-99) will be integer variables	All the elements (0-99) will be referred as a common name rollno

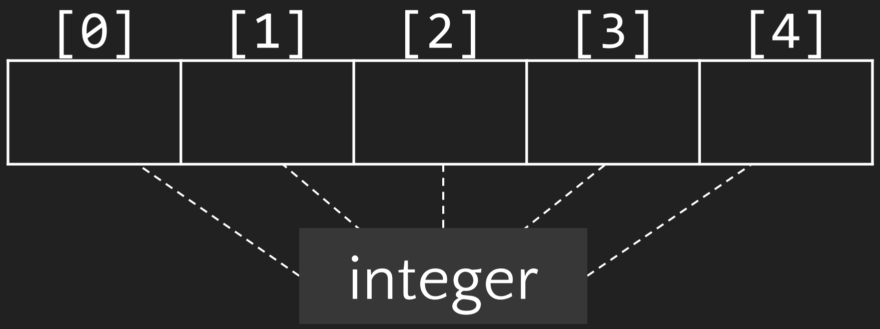
Declaring an array

Syntax

```
data-type variable-name[size];
```

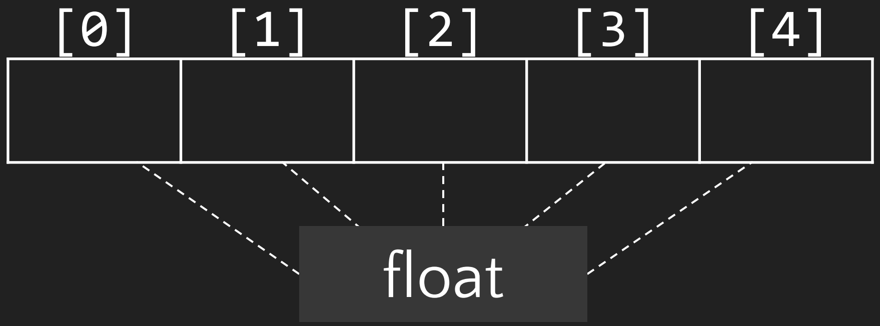
Integer Array

```
int mark[5];
```



Float Array

```
float avg[5];
```



- By default array index starts with `0`.
- If we declare an array of size 5 then its index ranges from `0` to 4.
- First element will be store at `mark[0]` and last element will be stored at `mark[4]` not `mark[5]`.
- Like integer and float array we can declare array of type `char`.

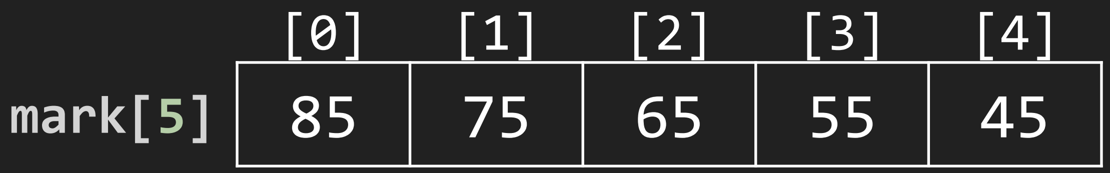
Initialing and Accessing an Array

Declaring, initializing and accessing single integer variable

```c
int mark=90;      //variable mark is initialized with value 90
printf("%d",mark); //mark value printed
```

Declaring, initializing and accessing integer array variable

```c
int mark[5]={85,75,76,55,45}; //mark is initialized with 5 values
printf("%d",mark[0]); //prints 85
printf("%d",mark[1]); //prints 75
printf("%d",mark[2]); //prints 65
printf("%d",mark[3]); //prints 55
printf("%d",mark[4]); //prints 45
```



Read(Scan) Array Elements

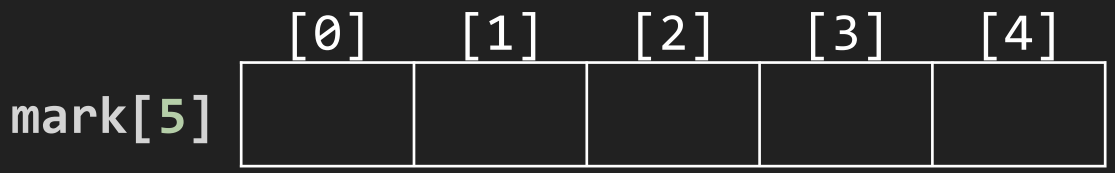
Reading array without loop

```
void main()
{
  int mark[5];
  printf("Enter array element=");
  scanf("%d",&mark[0]);
  printf("Enter array element=");
  scanf("%d",&mark[1]);
  printf("Enter array element=");
  scanf("%d",&mark[2]);
  printf("Enter array element=");
  scanf("%d",&mark[3]);
  printf("Enter array element=");
  scanf("%d",&mark[4]);

  printf("%d",mark[0]);
  printf("%d",mark[1]);
  printf("%d",mark[2]);
  printf("%d",mark[3]);
  printf("%d",mark[4]);
}
```

Reading array using loop

```
void main()
{
  int mark[5],i;
  for(i=0;i<5;i++)
  {
   printf("Enter array element=");
   scanf("%d",&mark[i]);
  }
  for(i=0;i<5;i++)
  {
   printf("%d",mark[i]);
  }
}
```



Develop a program to count number of positive or negative number from an array of 10 numbers.

Program

```c
void main(){
    int num[10],i,pos,neg;
    pos = 0;
    neg = 0;
    for(i=0;i<10;i++)
    {
        printf("Enter array element=");
        scanf("%d",&num[i]);
    }
    for(i=0;i<10;i++)
    {
        if(num[i]>0)
            pos=pos+1;
        else
            neg=neg+1;
    }
    printf("Positive=%d,Negative=%d",pos,neg);
}
```

Output

```
Enter array element=1
Enter array element=2
Enter array element=3
Enter array element=4
Enter array element=5
Enter array element=-1
Enter array element=-2
Enter array element=3
Enter array element=4
Enter array element=5
Positive=8,Negative=2
```

Develop a program to read n numbers in an array and print them in reverse order.

Program

```c
void main()
{
    int num[100],n,i;
    printf("Enter number of array elements=");
    scanf("%d",&n);
//loop will scan n elements only
    for(i=0;i<n;i++)
    {
        printf("Enter array element=");
        scanf("%d",&num[i]);
    }
//negative loop to print array in reverse order
    for(i=n-1;i>=0;i--)
    {
        printf("%d\n",num[i]);
    }
}
```

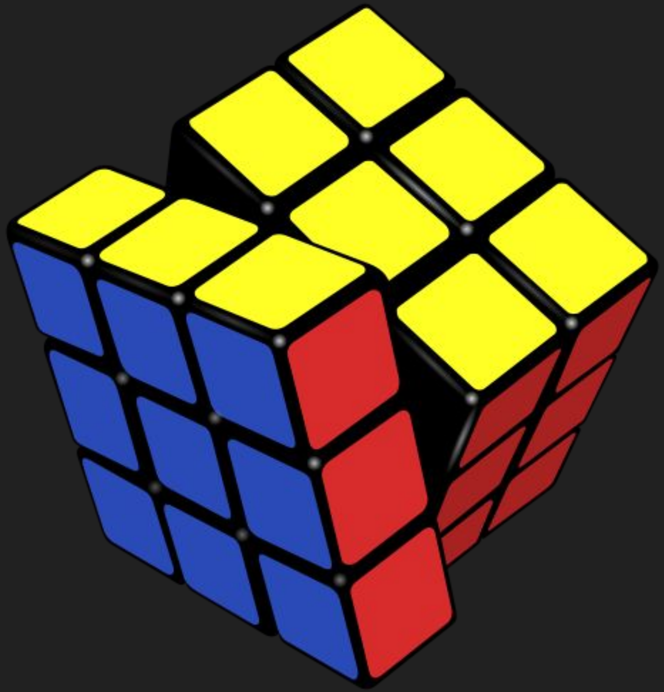
Output

```
Enter number of array
elements=5
Enter array element=1
Enter array element=2
Enter array element=3
Enter array element=4
Enter array element=5
5
4
3
2
1
```

Practice Programs

1) Develop a program to calculate sum of n array elements in C.
2) Develop a program to calculate average of n array elements in C.
3) Develop a program to find largest array element in C.
4) Develop a program to print sum of second and second last element of an array.
5) Develop a program to copy array elements to another array.
6) Develop a program to count odd and even elements of an array.

Multi Dimensional Array



Declaring 2 Dimensional Array

Syntax

```
data-type variable-name[x][y];
```

Declaration

```
int data[3][3]; //This array can hold 9 elements
```

```
int data[3][3];
```

	Column-0	Column-1	Column-2
Row-0	data[0][0]	data[0][1]	data[0][2]
Row-1	data[1][0]	data[1][1]	data[1][2]
Row-2	data[2][0]	data[2][1]	data[2][2]

- A two dimensional array can be seen as a table with 'x' rows and 'y' columns.
- The row number ranges from `0` to `(x-1)` and column number ranges from `0` to `(y-1)`.

Initialing and Accessing a 2D Array: Example-1

Program

```c
int data[3][3] = {
{1,2,3}, //row 0 with 3 elements
{4,5,6}, //row 1 with 3 elements
{7,8,9}  //row 2 with 3 elements
    };
printf("%d",data[0][0]); //1
printf("%d",data[0][1]); //2
printf("%d\n",data[0][2]); //3

printf("%d",data[1][0]); //4
printf("%d",data[1][1]);  //5
printf("%d\n",data[1][2]);  //6

printf("%d",data[2][0]);//7
printf("%d",data[2][1]); //8
printf("%d",data[2][2]); //9
```

```c
// data[3][3] can be initialized like this also
int data[3][3]={{1,2,3},{4,5,6},{7,8,9}};
```

	Column-0	Column-1	Column-2
Row-0	1	2	3
Row-1	4	5	6
Row-2	7	8	9

Initialing and Accessing a 2D Array: Example-2

Program

```c
int data[2][4] = {
{1,2,3,4}, //row 0 with 4 elements
{5,6,7,8}, //row 1 with 4 elements
    };
printf("%d",data[0][0]); //1
printf("%d",data[0][1]); //2
printf("%d",data[0][2]); //3
printf("%d\n",data[0][3]); //4

printf("%d",data[1][0]); //5
printf("%d",data[1][1]); //6
printf("%d",data[1][2]); //7
printf("%d",data[1][3]); //8
```

```c
// data[2][4] can be initialized like this also
int data[2][4]={{1,2,3,4},{5,6,7,8}};
```

	Col-0	Col-1	Col-2	Col-3
Row-0	1	2	3	4
Row-1	5	6	7	8

Read(Scan) 2D Array Elements

Program

```c
void main(){
    int data[3][3],i,j;
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("Enter array element=");
            scanf("%d",&data[i][j]);
        }
    }
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("%d",data[i][j]);
        }
        printf("\n");
    }
}
```

	Column-0	Column-1	Column-2
Row-0	1	2	3
Row-1	4	5	6
Row-2	7	8	9

Output

```
Enter array element=1
Enter array element=2
Enter array element=3
Enter array element=4
Enter array element=5
Enter array element=6
Enter array element=7
Enter array element=8
Enter array element=9
123
456
789
```

Develop a program to count number of positive, negative and zero elements from 3 X 3 matrix

Program

```c
void main(){
    int data[3][3],i,j,pos=0,neg=0,zero=0;
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("Enter array element=");
            scanf("%d",&data[i][j]);
            if(data[i][j]>0)
                pos=pos+1;
            else if(data[i][j]<0)
                neg=neg+1;
            else
                zero=zero+1;
        }
    }
    printf("positive=%d,negative=%d,zero=%d",pos,neg,zero);
}
```

Output

```
Enter array element=9
Enter array element=5
Enter array element=6
Enter array element=-3
Enter array element=-7
Enter array element=0
Enter array element=11
Enter array element=13
Enter array element=8
positive=6,negative=2,zero=1
```

Practice Programs

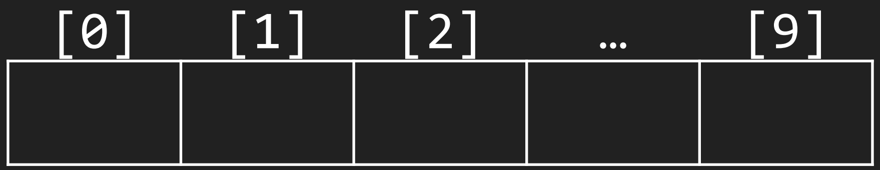
1. Develop a program to perform addition of two matrix.
2. Develop a program to perform multiplication of two matrix.

String
(Character Array)

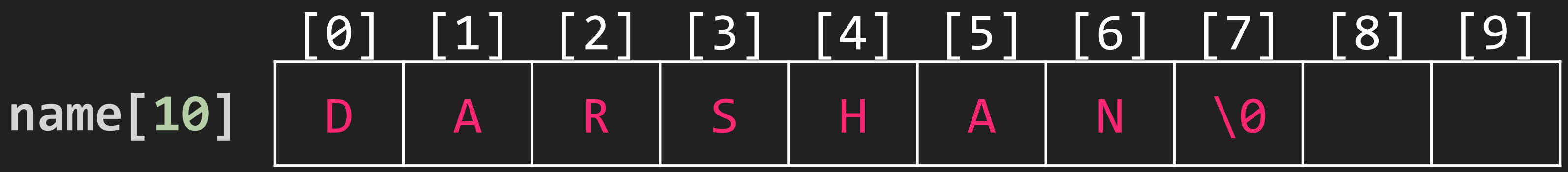
Definition: String

- A String is a one-dimensional array of characters terminated by a `null('\0')`.

```c
char name[10];
```



- Each character in the array occupies one byte of memory, and the last character must always be `null('\0')`.
- The termination character `('\0')` is important in a string to identify where the string ends.



Declaring & Initializing String

Declaration

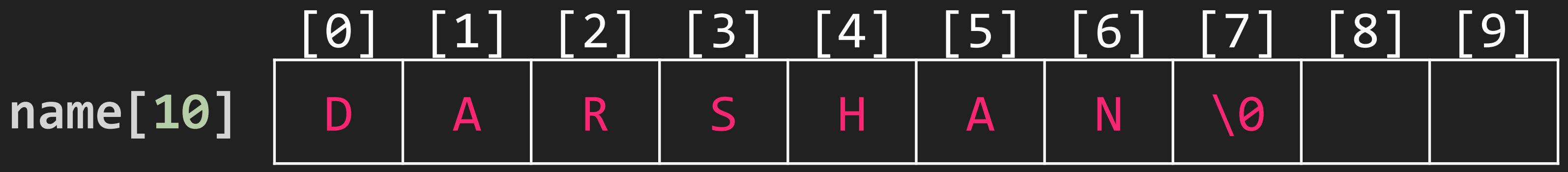
```c
char name[10];
```

Initialization method 1:

```c
char name[10]={'D','A','R','S','H','A','N','\0'};
```

Initialization method 2:

```c
char name[10]="DARSHAN";
//'\0' will be automatically inserted at the end in this type of declaration.
```



Read String: scanf()

Program

```c
void main()
{
	char name[10];
	printf("Enter name:");
	scanf("%s",name);
	printf("Name=%s",name);
}
```

Output

```
Enter name: Darshan
Name=Darshan
```

Output

```
Enter name: CE Darshan
Name=CE
```

- There is no need to use address of `(&)` operator in `scanf` to store a string.
- As string name is an array of characters and the name of the array, i.e., name indicates the base address of the string (character array).
- `scanf()` terminates its input on the first whitespace(space, tab, newline etc.) encountered.

Read String: gets()

Program

```c
#include<stdio.h>
void main()
{
    char name[10];
    printf("Enter name:");
    gets(name); //read string including white spaces
    printf("Name=%s",name);
}
```

Output

```
Enter name:Darshan Institute
Name=Darshan Institute
```

- gets(): Reads characters from the standard input and stores them as a string.
- puts(): Prints characters from the standard.
- scanf(): Reads input until it encounters whitespace, newline or End Of File(EOF) whereas gets() reads input until it encounters newline or End Of File(EOF).
- gets(): Does not stop reading input when it encounters whitespace instead it takes whitespace as a string.

String Handling Functions : strlen()

- C has several inbuilt functions to operate on string. These functions are known as string handling functions.
- `strlen(s1):` returns length of a string in integer

Program

```c
#include <stdio.h>
#include <string.h> //header file for string functions
void main()
{
    char s1[10];
    printf("Enter string:");
    gets(s1);
    printf("%d",strlen(s1)); // returns length of s1 in integer
}
```

Output

```
Enter string: CE Darshan
10
```

String Handling Functions: strcmp()

- `strcmp(s1,s2)`: Returns `0` if `s1` and `s2` are the same.
- Returns less than `0` if `s1<s2`.
- Returns greater than `0` if `s1>s2`.

Program

```c
void main()
{
    char s1[10],s2[10];
    printf("Enter string-1:");
    gets(s1);
    printf("Enter string-2:");
    gets(s2);
    if(strcmp(s1,s2)==0)
       printf("Strings are same");
    else
       printf("Strings are not same");
}
```

Output

```
Enter string-1:Computer
Enter string-2:Computer
Strings are same
```

Output

```
Enter string-1:Computer
Enter string-2:Computer
Strings are same
```

String Handling Functions

For examples consider: `char s1[]="Their",s2[]="There";`

Syntax	Description
`strcpy(s1,s2)`	Copies 2nd string to 1st string. `strcpy(s1,s2)` copies the string `s2` in to string `s1` so `s1` is now “There”. `s2` remains unchanged.
`strcat(s1,s2)`	Appends 2nd string at the end of 1st string. `strcat(s1,s2);` a copy of string `s2` is appended at the end of string `s1`. Now `s1` becomes “TheirThere”
`strchr(s1,c)`	Returns a pointer to the first occurrence of a given character in the string `s1`. `printf("%s",strchr(s1,'i'));` Output : ir
`strstr(s1,s2)`	Returns a pointer to the first occurrence of a given string `s2` in string `s1`. `printf("%s",strstr(s1,"he"));` Output : heir

String Handling Functions (Cont...)

For examples consider: `char s1[]="Their",s2[]="There";`

Syntax	Description
`strrev(s1)`	Reverses given string. `strrev(s1);` makes string `s1` to "riehT"
`strlwr(s1)`	Converts string `s1` to lower case. `printf("%s",strlwr(s1));` Output : their
`strupr(s1)`	Converts string `s1` to upper case. `printf("%s",strupr(s1));` Output : THEIR

`strncpy(s1,s2,n)`	Copies first n character of string `s2` to string `s1`
	`s1=""; s2="There";`
	`strncpy(s1,s2,2);`
	`printf("%s",s1);` Output : Th

`strncat(s1,s2,n)`	Appends first n character of string `s2` at the end of string `s1`. `strncat(s1,s2,2);` `printf("%s", s1);` Output : TheirTh

String Handling Functions (Cont...)

For examples consider: `char s1[]="Their",s2[]="There";`

Syntax	Description
`strncmp(s1,s2,n)`	Compares first `n` character of string `s1` and `s2` and returns similar result as `strcmp()` function. `printf("%d",strcmp(s1,s2,3));` Output : 0
`strrchr(s1,c)`	Returns the last occurrence of a given character in a string `s1`. `printf("%s",strrchr(s2,'e'));` Output : ere

Thank you