* Char Arrays and strings:

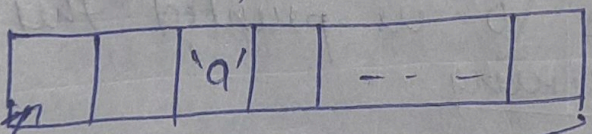
char datatype → 1 byte

Range of char datatype → $-2^7 - 2^7 - 1$

$-128 \rightarrow 127$ (signed)

$0 \rightarrow 2^8 - 1$ (unsigned)

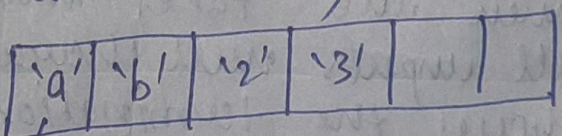
int arr[10];



we can store integer type value.

can we store 'a' yes as it will store ascii value of a i.e 97

char arr[10];



we can store character type value.

we can also store integer with single quotes to use it as a character.

~~Inputs~~

* How to take input in char datatype array.

In integer

```
int arr[10];
cin >> arr[i];
```

```
int x;
cin >> x;
```

Taking input in char datatype by single character

```
char ch[100]; → char array
ch[0] = 's';
ch[1] = 'u';
ch[2] = 'b';
cin >> ch[3] >> ch[4] >> ch[5];
for(i=0; i<6; i++){
    cout << ch[i];
}
```

output

subrat

↳ input

while print → subrat

We can also take input multiple inputs in char array

```
char name[100];
cin >> name;
cout << name;
```

But what if I take inputs with spaces.

I/P subrat

O/P subrat

0	1	2	3	4	5	6
s	u	b	h	a	t	\0

↑ P

when we take multiple characters inputs then after all inputs null character i.e \0 is printed this shows the termination of string

sequence of characters.

Space complexity: O(1).

* De-merits of in while char datatype:

cin can't recognize → space, enter (tab) or '\n'

ex: ~~char~~ char name[20];
cin >> name;
cout << name;

I/P : subrat	I/P : subrat singh
O/P : subrat	O/P : subrat

(space)

→ cin.getline(name , 50);
(name: variable name; 50: capacity of an array)

it is used to take char/string input with spaces. it will consider whole string with space.

can access char & strings type input along with the space

→ ~~st~~.

① Length of a string:

```
int getlength(char arr[]){
    int i=0;
    int length=0;
    while(arr[i]!='\0'){
        length++;
        i++;
    }
    return length;
}
int main(){
    char arr[100];
    cout<<"Enter the name:";
    cin.getline(arr,100);
    int length = getlength(arr);
    cout<<"The length is:"<<length
```

strlen(arr) func will also give the length of a string. we needed file
#include<string.h>

T.C = O(n)
S.C = O(1)

output: Subrat Kumar Singh
The length is 18.

② Reverse a string

T.C = O(n)
S.C = O(1)

```
void reverseTheString (char arr[]){
    int i=0;
    int j= strlen(arr)-1;
    while (i<=j){
        swap (arr[i], arr[j]);
        i++;
        j--;
    }
}

int main(){
    char arr[100];
    cout << "Enter the name:";
    cin.getline (arr, 50);
    reverseTheString (arr);
    cout << "The Reverse name is :" << arr;
```

output: Enter the name: Subrat Kumar Singh
The reverse name: hgnis ramuK tarbuS

③ Replace all spaces by @:

```
void replaceTheSpaces (char sentence []){
    for (int i=0; i< strlen (sentence); i++){
        if (sentence [i] == ' '){
            sentence [i] = '@';
        }
    }
}

int main (){
    char sentence [100];
    cout << "Enter any sentence :";
    cin.getline (sentence, 100);
    ~~cout << "The updated sentence is :" <<~~
    replaceTheSpaces (sentence);
    cout << "The updated sentence is :" << sentence;
}
```

T.C = O(n)
S.C = O(1)

output: My name is Khan
My @name@is@Khan.

④ Palindrome:

→ Noon → Read left to right & right to left it will be same.

~~Character~~
Racecar → Read left to right & right to left it will be same.

```
bool checkPalindrome (char word []){
    int i=0;
    int j= strlen(word)-1;
    while (i<=j){
        if (word[i]==word[j]){
            return true;
            i++;
            j--;
        }
        else{
            return false;
        }
    }
}
```

```
int main(){
    char word[100];
    cout<<"Enter the word:";
    cin.getline(word, 100);
    if (checkPalindrome(word)){
        cout<<"It is a Palindrome";
    }
    else{
        cout<<"It is not a Palindrom
}}
```

Output: noon
It is a Palindrome
racecar
It is a Palindrome
subrat
It is not a Palindrome.

(5) Uppercase the string

```
void makeUppercase (char word[]){
    for(int i=0; i<strlen(word); i++)
    {word[i] = word[i] - 32;
    }
}
int main(){
cout<<"Enter the word";
cin.getline(word, 100);
makeUppercase(word);
cout<<"The word in Uppercase
    is:"<<word;
}
```

(6) Lowercase the string

```
void makeLowercase(char
                   word[])
{
for(int i=0; i<strlen(word)
                ; i++){
word[i] = word[i]+32;
}
}
```

same main fune

Output

subrat
SUBRAT

SUBRAT
subrat

* Strings: → It is a sequence of characters..
→ It is a dynamic datatype for char array.
→ same alike of vector. We don't need to explicity give the size of string.

→ Strings have some inbuilt functions. which we can use along with the strings.

→ syntax of string is `string s = "Hello";`
(string: datatype, s: variable name)

→ Inbuit Functions:

(1) length() → will give the length of string.
ex: string s = "Hello";
s.length() = 5
| size(): too have same func of length() can use any.

(2) capacity() → max value can a string have.
ex: string s = "Test String";
s.capacity() = 15.

(3) empty() → To check string is empty or not.
ex: string s = "Subrat";
s.empty() = 0/No
1/Yes.

(4) push-back() → To add character to any a string,
ex: string s = "subrat";
s.push-back = 'a';
Output: subrata.

(5) pop-back() → To delete last character;
ex: string s = "subrat"
s.pop-back();
subra.

⑥ substr(): → to generate substring.

ex: string word = "Subrat is good";

word.substr(10, 4)

from index 10 and print 4 characters.

output good..

⑦ find(): → to find content in string.

ex: string s = "Hello world";

s.find("world");

found at index 6.

⑧ npos → indicates the that the character or substring you are finding in string is not found.