Page No.
Date :

* Tokenization using 'strtok()':

→ we can use this inbuilt function to break the string in ~~to~~ a list of token

// char *strtok(char *s, char *delimeters)

string ↗ (above `*s`); abt which you want to break the string ↗ (above `*delimeters`)

eg
```
int main()
{
    char s[100]= "Today is a rainy day";

    char *ptr= strtok(s, " ");        → delimeter (above " ")
    cout<<ptr<<endl;   → Today

    ptr= strtok(s, " ");
     cout<<ptr<<endl;   → Today
```

Now ~~for~~ to get other tokens

// on the first call function should be passed with string argument for 's'.

// on subsequent calls we should pass the string argument as null.

↳ when you pass NULL the strtok acctually maintains a static variable/array that stores the state of the string.

→ it maintains that I have already ~~cone~~ covered this part of the string

So, int main()
{
    char s[100] = "Today is a rainy day";
    char *ptr = strtok(s, " ");

    cout<<ptr<<endl; → Today
    ptr = strtok (NULL, " ");
    cout <<ptr<<endl; → is
}

Thus, to get all values

```
int main()
{
    char s[100] = "Today is a rainy day";
    char *ptr = strtok(s, " ");
    cout << ptr << endl;
    ~~ptr = strtok(NULL, " ");~~
    while (ptr!= NULL) {
        ptr = strtok(NULL, " ");
        cout << ptr <endl;
    }
    return 0;
}
```

last wala NULL

Page No.
Date :

it will accept as well as return char.

⇒ Remember :

this must be char * so if 's' is string then you'll write

char *pp = strtok(s, " ")

strtok((char*) s. c-str(), " ");

we can push-back pp to any string vector of vector<string> result; result. push-back(pp);

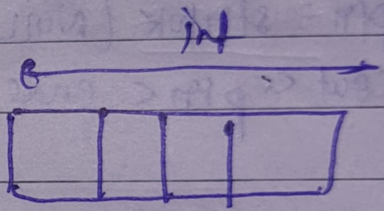
strtok((char *) s. c-str(), " ");

pointers

* Address of operator '&'

```
int x = 10 ;
cout << &x << endl ;
```

C++ display address in hexadecimal form.



8080

Base 16

0x ff de 68

0, 1, 2, 3 . . . 9, A B . . . F.

exception it doesn't work for character variables

```
char ch = 'A' ;
cout << &ch << endl ;
```
→ it will print 'A'.

it happen because of operator overloading because of $\boxed{<<}$ this operator.