Expt. No. 13 Page No. 1

# STRING

string > character arrays

# compare with character array

| Character Arrays | Strings |
|---|---|
| • Need to know size beforehand | • No need to know size beforehand |
| • Larger size required for operations (concatenate or append). | • Performing operations like concatenation & append is easier. |
| • No terminating extra character | • Terminated with a special character '\0' |

# Required header

```
#include <string>          // have to use namespace std
using namespace std;       // use this to avoid namespace error
                              or use 'std::string'
```

# Declaration

```
string str = "This is string";
string str(5, 'N');        // NNNNN
string str0(str);          // make copy of str in str1
```

Teacher's Signature : _____

# Input Output

Input

```
cin >> str;          // this will input only first word
getline (cin, str)   // this will input one sentense
```

output

```
cout << str << endl;
```

# Different Function of String

1. Append :- `str.append("some string")`
   `str = str + "some string"`
   `str += "some string"`

   Time Complexity = O(N)  N = size of new string

2. assign :- `str.assign("new string")`
   `str = "new string"`

3. at :- `str.at(INDEX)`
   `str[INDEX]`

   Time complexity = O(1)

4. begin :- Returns iterator pointing to first character

   Time complexity = O(1)

```
auto i = str.begin()   // data type : iterator
```

Teacher's Signature : _____

5. clear :- str.clear() // clear and assign to ("") string of length zero
Time complexity = O(1)

6. Compare :- s2.compare(s1) will integer
>0 if s2 is greater than s1
0 if s1 = s2
<0 if s2 is smaller than s1

Time complexity = O(M+N) M & N are sizes of strings s1 & s2

7. c-str :- str.c_str()
converts to c style string & returns pointer to c style string.
Time complexity = O(1)

8. empty :- str.empty() returns true if string is empty other wise false
Time complexity = O(1)

9. end :- str.end() returns iterator to next to last ~~of end~~ character.
Time complexity = O(1)

10. erase :- str.erase(position, range) remove range characters from position in string
Time complexity = O(N)
N = size of new string

Teacher's Signature : _____

11. find :- str.find("string") ~~returns of param~~
returns first occurrence of the parameter in the string otherwise returns very large value
Time complexity = O(N) ; N = size of string.

12. insert :- str.insert(position, string)
insert 'string' to 'position' in 'str'
Time complexity = O(N) ; N = size of new string

13. length :- str.length() returns size of string
Time complexity = O(1)

14. resize :- resize to small or bigger length
str.resize(newsize)
Time complexity = O(N) ; N = size of new string

15. size :- str.size() returns size of string
Time complexity = O(1)

16. substr :- string s = str.substr(position, length)
returns substring of length 'length' from position 'position'
Time complexity = O(N) ; N is size of substring

17. stoi :- int x = stoi(str)
returns string converted to 'int'

Teacher's Signature : _____

NOTE :-

1. to_string() :- convert int to string

string str = to_string(x)

2. Sorting a string :-

* header file required : <algorithm>

* function to use : sort()

sort ( begin-iterator , end-iterator)

code:

```
string s = "HITESH";
sort(s.begin() , s.end());
cout << s <<endl;          // EHHIST
```

Teacher's Signature : _____