Operations on strings

Input Functions

1. getline() :- This function is used to **store a stream of characters** as entered by the user in the object memory.

2. push_back() :- This function is used to **input** a character at the **end** of the string.

3. pop_back() :- Introduced from C++11(for strings), this function is used to **delete the last character** from the string.

```cpp
// C++ code to demonstrate the working of
// getline(), push_back() and pop_back()
#include<iostream>
#include<string> // for string class
using namespace std;
int main()
{
    // Declaring string
    string str;

    // Taking string input using getline()
    // "geeksforgeek" in givin output
    getline(cin,str);

    // Displaying string
    cout << "The initial string is : ";
    cout << str << endl;

    // Using push_back() to insert a character
    // at end
    // pushes 's' in this case
    str.push_back('s');

    // Displaying string
    cout << "The string after push_back operation is : ";
    cout << str << endl;

     // Using pop_back() to delete a character
     // from end
     // pops 's' in this case
     str.pop_back();

     // Displaying string
     cout << "The string after pop_back operation is : ";
     cout << str << endl;

     return 0;

  }
```

Input:

```
geeksforgeek
```

Output:

```
The initial string is : geeksforgeek
The string after push_back operation is : geeksforgeeks
The string after pop_back operation is : geeksforgeek
```

Capacity Functions

3. capacity() :- This function **returns the capacity** allocated to the string, which can be **equal to or more than the size** of the string. Additional space is allocated so that when the new characters are added to the string, the **operations can be done efficiently**.

4. resize() :- This function **changes the size of string**, the size can be increased or decreased.

5.shrink_to_fit() :- This function **decreases the capacity** of the string and makes it equal to its size. This operation is **useful to save additional memory** if we are sure that no further addition of characters have to be made.

```cpp
// C++ code to demonstrate the working of
// capacity(), resize() and shrink_to_fit()
#include<iostream>
#include<string> // for string class
using namespace std;
int main()
{
    // Initializing string
    string str = "geeksforgeeks is for geeks";

    // Displaying string
    cout << "The initial string is : ";
    cout << str << endl;

    // Resizing string using resize()
    str.resize(13);

    // Displaying string
    cout << "The string after resize operation is : ";
    cout << str << endl;

    // Displaying capacity of string
    cout << "The capacity of string is : ";
    cout << str.capacity() << endl;

    // Decreasing the capacity of string
    // using shrink_to_fit()
    str.shrink_to_fit();
    // Displaying string
    cout << "The new capacity after shrinking is : ";
    cout << str.capacity() << endl;

    return 0;

}
```

Output:

```
The initial string is : geeksforgeeks is for geeks
The string after resize operation is : geeksforgeeks
The capacity of string is : 26
The new capacity after shrinking is : 13
```

Iterator Functions

7. begin() :- This function returns an **iterator** to **beginning** of the string.

8. end() :- This function returns an **iterator** to **end** of the string.

9. rbegin() :- This function returns a **reverse iterator** pointing at the **end** of string.

10. rend() :- This function returns a **reverse iterator** pointing at **beginning** of string.

```
// C++ code to demonstrate the working of
// begin(), end(), rbegin(), rend()
#include<iostream>
#include<string> // for string class
using namespace std;
int main()
{
    // Initializing string`
    string str = "geeksforgeeks";

    // Declaring iterator
    std::string::iterator it;

    // Declaring reverse iterator
    std::string::reverse_iterator it1;

// Displaying string
cout << "The string using forward iterators is : ";
for (it=str.begin(); it!=str.end(); it++)
cout << *it;
cout << endl;
```

```
    // Displaying reverse string
    cout << "The reverse string using reverse iterators is : ";
    for (it1=str.rbegin(); it1!=str.rend(); it1++)
    cout << *it1;
    cout << endl;

    return 0;

}
```

Output:

```
The string using forward iterators is : geeksforgeeks
The reverse string using reverse iterators is : skeegrofskeeg
```

Manipulating Functions

11. copy("char array", len, pos) :- This function **copies the substring in target character array** mentioned in its arguments. It takes 3 arguments, **target char array, length to be copied and starting position in string to start copying.**

12. swap() :- This function **swaps** one string with other.

```
// C++ code to demonstrate the working of
// copy() and swap()
#include<iostream>
#include<string> // for string class
using namespace std;
int main()
{
    // Initializing 1st string
    string str1 = "geeksforgeeks is for geeks";

    // Declaring 2nd string
    string str2 = "geeksforgeeks rocks";

    // Declaring character array
    char ch[80];

    // using copy() to copy elements into char array
    // copies "geeksforgeeks"
    str1.copy(ch,13,0);

    // Diplaying char array
    cout << "The new copied character array is : ";
    cout << ch << endl << endl;

    // Displaying strings before swapping
    cout << "The 1st string before swapping is : ";
    cout << str1 << endl;
    cout << "The 2nd string before swapping is : ";
    cout << str2 << endl;
```

Output:

```
The new copied character array is : geeksforgeeks

The 1st string before swapping is : geeksforgeeks is for geeks
The 2nd string before swapping is : geeksforgeeks rocks
The 1st string after swapping is : geeksforgeeks rocks
The 2nd string after swapping is : geeksforgeeks is for geeks
```

}