Difference between concatenation of strings

```
str += s
```

vs

```
str = str + s
```

(MUST READ)

@Chandanagrawal23

Check problem here : [Removing Stars From a String - LeetCode]

```cpp
stack <char> stk; // stack contains "chandan"
string ans = "";
while(!stk.empty())
{
    ans += stk.top();// same as ans.push_back(stk.top());
    stk.pop();
}
reverse(ans.begin(),ans.end());
```

```cpp
stack <char> stk; // stack contains "chandan"
string ans = "";
while(!stk.empty())
{
    ans = ans + stk.top();
    stk.pop();
}
reverse(ans.begin(),ans.end());
```

+= operator is much faster than **+** operator (more than 70% faster).

Program will give **MLE/TLE** error while using **+** operator

08/29/2022 14:12	Memory Limit Exceeded **using + operator**	N/A	N/A	cpp
08/29/2022 14:11	Accepted **using += operator**	108 ms	27.1 MB	cpp
08/28/2022 08:09	Accepted	159 ms	27.1 MB	cpp
08/28/2022 08:07	Time Limit Exceeded	N/A	N/A	cpp

But WHY???

@Chandanagrawal23

Reason:

Let's go to the definition of **+=** operator and **+** operator –

std::string has members **operator +** and **operator +=**

The former is usually implemented with the latter by way of an intermediate temporary.

```
/// note reference return type
std::string& operator += (char c)
{
   this->append(c);
   return *this;
}
```

```
// note value return type
std::string operator + (char c) const
{
   std::string temp = *this;
   temp += c; // or just temp.append(c) directly
   return temp;
}
```

In the case of the addition assignment operator (**+=**) the character is directly added to **str** while in case of addition operator (**+**) the old string is first assigned to **temp** string and then the new char is added to **temp** and return it.

Look below to code snippets

```
stack <char> stk; // stack contains "chandan"
string ans = "";
while(!stk.empty())
{
    ans += stk.top();// same as ans.push_back(stk.top());
    stk.pop();
}
reverse(ans.begin(),ans.end());
```

```
stack <char> stk; // stack contains "chandan"
string ans = "";
while(!stk.empty())
{
    ans = stk.top() + ans;
    stk.pop();
}
```

In this first code what I did is **first add every character from stack to string and after that I reversed whole string.**

While in the second one, **I add a character from the stack into the beginning of the string**.

Now the same issue happens, the second one gives me TLE/MLE (reason is same as above).

08/30/2022 14:27	Memory Limit Exceeded	N/A	N/A	cpp
08/30/2022 14:26	Accepted	142 ms	26.9 MB	cpp

So, make sure whenever you get this type of situation always use-

- **str += s[i]**
- **str.push_back(s[i])**
- **first push all char's and then reverse (if required).**

THANKS

@Chandanagrawal23