Part I: Practice and Theory

The following problems are for practise only and will **not be collected**.

Review problems: All. **Practice Problems**: P10.1, P10.2, P10.5, P10.9.

Part II: Programming. The following problems will be **collected** and two of them graded. Read instructions carefully!

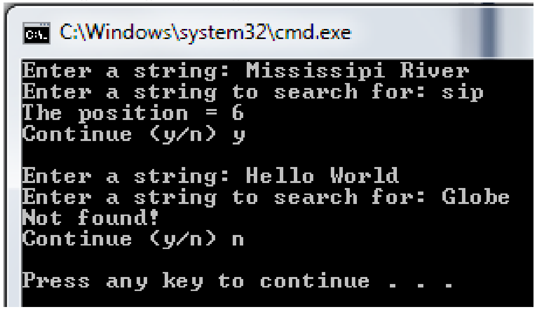
(1) **Problem P10.6**

- Implement the function

```
int index_of(string s, string t)
```

described in Problem P10.6 using **recursion**.

- Write a program that prompts the user to enter a string `s`, and and then a string `t`. Then use the function `index_of` to compute the index of the starting position of the first substring of the string `s` that matches `t`. Display the position if such position is found. If not found, display 'Not found!'.
- Implement a loop that keeps requesting the two strings until the user requests to quit.
- Submit the solution as `hmw_4_1.cpp`.
- Sample input-output:



```
Enter a string: Mississipi River
Enter a string to search for: sip
The position = 6
Continue (y/n) y

Enter a string: Hello World
Enter a string to search for: Globe
Not found!
Continue (y/n) n

Press any key to continue . . .
```

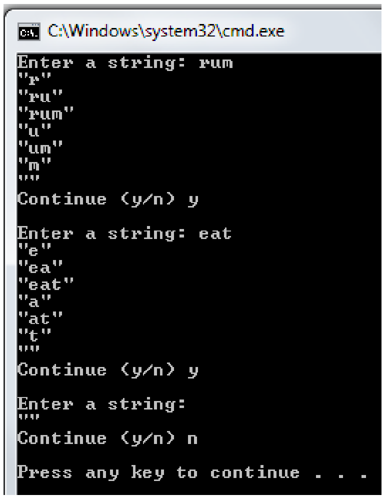
(2) - **Problem P10.10**

- Implement the function

```
vector <string> generate_substrings(string s)
```

described in Problem P10.10 using **recursion**.

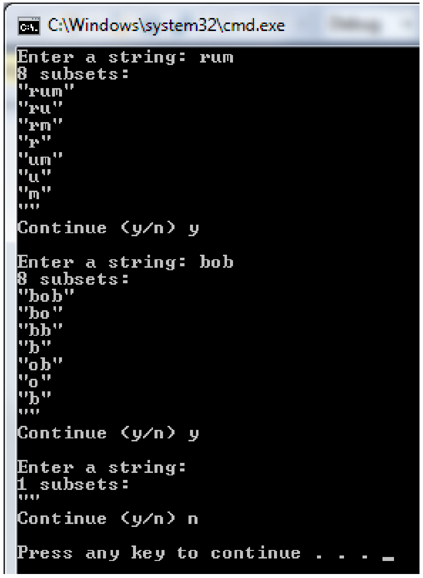
- Write a program that prompts the user to enter a string `s`. Then generate all substrings of the string `s` using the function `generate_substrings`. Then display all the substrings (one substring per line).
- Note, an empty string is a substring of any string and hence it must be a part of the output.
- Submit the solution as `hmw_4_2.cpp`.
- Sample input-output:



```
Enter a string: rum
"r"
"ru"
"rum"
"u"
"um"
"m"
""
Continue (y/n) y

Enter a string: eat
"e"
"ea"
"eat"
"a"
"at"
"t"
""
Continue (y/n) y

Enter a string:
""
Continue (y/n) n

Press any key to continue . . .
```

(3) **Problem P10.11**

- Implement the function `vector <string> generate_subsets(string s)` described in Problem P10.11 using **recursion**.
- Write a program that prompts the user to enter a string `s`. Then generates all subsets of characters of the string `s` and display them (a subset per line).
- Note: a string `s` can contain letters that are repeated, say `s`="bob". The algorithm should treat all letters as individual 'place holders' without taking into account its content. Thus, the number of subsets of characters for a string containing n characters must be 2^n.
- Note: an empty string is a subset of any string and hence it must be a part of the output.
- Submit the solution as `hmw_4_3.cpp`.
- Sample input-output:



```
Enter a string: rum
8 subsets:
"rum"
"ru"
"rm"
"r"
"um"
"u"
"m"
""
Continue (y/n) y

Enter a string: bob
8 subsets:
"bob"
"bo"
"bb"
"b"
"ob"
"o"
"b"
""
Continue (y/n) y

Enter a string:
1 subsets:
""
Continue (y/n) n

Press any key to continue . . .
```