第 5 回 C++ プログラミング II 実験

出題日：2021 年 5 月 7 日（金）15:00 JST
第一提出期限：2021 年 5 月 7 日（金）16:40 JST
第二提出期限：2021 年 5 月 13 日（木）23:59 JST
提出方法：Course Power の指定された箇所に提出

実験課題 5

Course Power から ファイル（prac05-1-main.cpp, prac05-2-main.cpp）をダウンロードして，以下の問題および実行例をもとに，プログラムを完成させなさい．

問題 (1)

deque を用いて，入力された文字列の履歴を最大 3 件まで保存するプログラムを作成せよ．

過去の入力ほど deque の先頭に保存されるものとし，履歴が 3 件を超えるときは最も過去の履歴が deque から削除されるものとする（つまり，deque の先頭の要素が削除される）．

実行時は，文字列を 5 回入力するものとする．また，入力するたびに保存された履歴を表示する機能を付けること．

実行例 1

```
./prac05-1.exe
Input: 1
history: 1
Input: a
history: 1 a
Input: hello
history: 1 a hello
Input: world
history: a hello world
Input: goodbye
history: hello world goodbye
```

問題 (2)

prac05-2-main.cpp は，ユーザが入力したコマンド（文字列）を表示するプログラムである．“exit” と入力することでプログラムは終了する．このプログラムに以下で説明する機能を追加し，プログラムを完成させよ．なお，prac05-2-main.cpp には文字列の入力や各 stack を表示する部分のプログラムが書かれているが，適宜書き換えて構わない．

履歴保存機能

入力されたコマンド（文字列）の履歴を最大 3 件まで保存する機能（問題 (1) と同じもの）．なお，後述の Undo 機能を作成するために，履歴は deque の undo_stack を利用して保存することを勧める（説明もその前提で書いている）．

Undo 機能

Undo 機能とは，直前の操作を取り消して過去の状態に戻す機能のことである．この課題では “undo” と入力することで，履歴（undo_stack）を 1 つ前の状態へ戻す機能として作成する．ただし，すでに古い履歴として削除された履歴については戻らないものとする．

Redo 機能

Redo 機能とは，Undo 機能によって取り消した操作をやり直す機能のことである．この課題では “redo” と入力することで，“undo” によって取り消されたコマンド（文字列）を再び実行する機能として作成する．

つまり，履歴（undo_stack の中身）が $\{1, 2, 3\}$ の状態であったとしたら，“undo” と入力することで履歴が $\{1, 2\}$ となる．このとき取り消された操作は「3 を入力したこと」であるから，“redo” と 入力することで「3 を入力したこと」が再度実行される（「3」が表示されて，履歴に 3 が追加される）．この Redo 機能を作成するためには，Undo 機能によって取り消したコマンド（文字列）の履歴を保存する必要があり，prac05-2-main.cpp では deque の redo_stack として用意されている．

Undo-Redo 機能のエラー対処（できる人だけやってみましょう）

1 度も入力履歴がない場合は Undo 機能を使うことができない．また，1 度も Undo 機能を使ったことがない場合は Redo 機能を使うことができない．これらを実行してしまうと，（おそらく）エラーでプログラムが終了する．そこでエラーで終了しないように，入力履歴や Undo の履歴を確認して，エラー終了しないよう対処せよ．

実行例 2

```
./prac05-2.exe
Input Command: 1
Current Command: [1]
+-------------------------------------+
undo_stack: 1
redo_stack:
+-------------------------------------+

Input Command: 2
Current Command: [2]
+-------------------------------------+
undo_stack: 1 2
redo_stack:
+-------------------------------------+

Input Command: 3
Current Command: [3]
+-------------------------------------+
undo_stack: 1 2 3
redo_stack:
+-------------------------------------+

Input Command: 4
Current Command: [4]
+-------------------------------------+
undo_stack: 2 3 4
redo_stack:
+-------------------------------------+
```

（つづき）

```
Input Command: undo
(Undo) Current Command: []
+-------------------------------------+
undo_stack: 2 3
redo_stack: 4
+-------------------------------------+

Input Command: undo
(Undo) Current Command: []
+-------------------------------------+
undo_stack: 2
redo_stack: 4 3
+-------------------------------------+

Input Command: redo
(Redo) Current Command: [3]
+-------------------------------------+
undo_stack: 2 3
redo_stack: 4
+-------------------------------------+

Input Command: redo
(Redo) Current Command: [4]
+-------------------------------------+
undo_stack: 2 3 4
redo_stack:
+-------------------------------------+

Input Command: exit
Exit.
```