# COSC2430 HW4: Queue

Kyle Cunningham

You need to create a C++ program that can decode messages by taking in commands and organizing them based on their priority.

In this assignment you will be given a few strings containing random characters and a list of instructions for how to decode the messages. The instructions are out of order and a priority queue must be used to initiate the commands in the proper order. When every command is put into the priority queue in the correct order and each command is performed then the program will decode a message.

The input will contain a series of commands, each command will be put into a priority queue. Each command is case sensitive. There are 6 possible commands:

- DECODE: which will be followed by a message inside of brackets
    - EX: DECODE:[dsacd# dsafdw](2)
    - This will add a string to be decoded to a second queue
- REPLACE: which will contain two characters separated by a
    - EX: REPLACE:[s,e](6)
    - Original string: slsphant
    - new string: elephant
    - This will replace all the characters with an ‘s’ to an ‘e’ in the top string of the message queue and then move that message to the end of the message queue
    - Has a priority of 6
- ADD:
    - EX: ADD:[n,a](4)
    - original string: bann
    - New string: banana
    - This will add an ‘a’ after every ‘n’ in the top string of the message queue and then move that message to the end of the message queue
    - And has a priority of 4
- REMOVE:
    - EX: REMOVE:[v](5)
    - Original string: mevssvavge
    - New string: message
    - This will remove every ‘v’ in the top string of the message queue and then move that message to the end of the message queue
    - Has a priority of 5
- SWAP:
    - EX: SWAP:[n,a](8)
    - Original string: bnanan

    - New string: banana
    - This will turn every ‘n’ into an ‘a’ and every ‘a’ into an ‘n’; in the top string of the message queue and then move that message to the end of the message queue
- PRINT:
    - EX: PRINT:(4)
    - This will print the value of the of the top string and take it out of the message queue

Each command will be followed by an integer from (1-10) that represents its priority in the queue. 1’s are the first in the queue and 10’s are the last. If two commands share the same number then the first one put into the queue goes first.

**In this program there are two queues that must be used:**

1. A priority queue that takes in every command sorting them based on priority
2. A regular queue that takes in the messages provided by the DECODE command

Every command besides the DECODE command will affect the top of the regular queue filled with messages, change the string, and then move that string to the back of the queue. So if there are two strings in the normal queue then they will be affected by every other command

The output will simply contain the decoded messages.

**Assumptions:**

1. The characters that it asks you to remove,replace,swap or add will never be commas or brackets (they might however be spaces though)
2. If the queue filled with messages to decode is empty and you are given any command other than the DECODE one then nothing should happen.
3. Each input will contain at maximum 1000 commands
4. The characters it asks you to add, replace or swap are case sensitive so if it asks you to replace t with b then don’t replace capital T’s

**Input41.txt**

```
REPLACE:[#,s](2)
DECODE:[?vi#zmf##agfzva#zbffnz4fco4f4](1)
REPLACE:[f,e](5)
REPLACE:[?,t](1)
PRINT:(7)
REPLACE:[z, ](4)
REPLACE:[4,d](6)
REPLACE:[v,h](3)
```

**Output41.txt**

```
this message has been decoded
```

Command line:
prioqueue input = input41.txt output = output41.txt

**Input42.txt**

```
REPLACE:[v,t](4)
SWAP:[s,e](6)
REMOVE:[l](3)
REPLACE:[e,%](7)
ADD:[g,e](7)
REPLACE:[r,#](7)
ADD:[R,e](4)
ADD:[f,r](6)
PRINT:(6)
ADD:[e,r](7)
ADD:[r,a](7)
ADD:[a,t](7)
REMOVE:[z](4)
DECODE:[k$eiHusg%$erzg%z$esy](1)
ADD:[t,o](7)
ADD:[o,r](7)
REPLACE:[#,r](8)
REPLACE:[g, ](2)
SWAP:[H,k](1)
ADD:[B,u](5)
REPLACE:[q, ](2)
ADD:[e,f](5)
DECODE:[Bvqelomsvimselqvhsyqdon'vqmaksqelsnels](1)
REPLACE:[$,a](5)
REMOVE:[%](2)
PRINT:(6)
DECODE:[R](3)
ADD:[r,i](6)
SWAP:[z,e](3)
ADD:[i,g](6)
REPLACE:[%,e](8)
PRINT:(9)
```

**output42.txt**

```
Haikus are easy
But sometimes they don’t make sense
Refrigerator
```

Command line:

prioqueue input = input42.txt output = output42.txt

**Input43.txt**

REPLACE:[q,a](3)
PRINT:(5)
REPLACE:[@,o](4)
SWAP:[z,e](1)
REPLACE:[u,e](5)
REPLACE:[!,e](2)
REPLACE:[x,t](4)
REMOVE:[4](6)
PRINT:(7)
DECODE:[This message never gets called by the print command](3)
DECODE:[7h15 M355463 w45 m05tly 9umb3r5](1)
REPLACE:[3,e](3)
SWAP:[a,b](4)
REPLACE:[7,T](2)
REPLACE:[5,s](4)
REPLACE:[#,m](2)
REPLACE:[&,a](3)
REPLACE:[9,n](1)
REPLACE:[$,s](4)
DECODE:[Thiz muzzqgu will only bu qvvucxud by luxxur xo luxxur ruplqcumunxz](2)
REPLACE:[v,f](3)
REPLACE:[0,o](1)
REPLACE:[z,s](4)
DECODE:[Th%$ @n! w&$ #&d! up @f @nly $y#b@l$](1)
REMOVE:[e](3)
REPLACE:[4,a](2)
REMOVE:[e](4)
REPLACE:[6,g](1)
PRINT:(4)
REPLACE:[%,i](2)
REPLACE:[1,i](3)

**Output43.txt**

This Message was mostly numbers
This one was made up of only symbols
This message will only be affected by letter to letter replacements

Command line:
prioqueue input = input41.txt output = output41.txt

**Visual Example:**
(this is just to show which string gets affected by which command
**input:**

REMOVE:[ ](4)
DECODE:[Second String](2)
SWAP:[h,k](1)
REPLACE:[n,5](3)
REMOVE:[4](9)
PRINT:(5)
DECODE:[String number one](1)
REMOVE:[g](2)
ADD:[2,g](1)
PRINT:(3)
SWAP:[e,S](2)

**This is what the priority queue should contain in order:**

SWAP:[h,k](1) *//this one happens before the DECODE commands can put*
*. //strings in the second queue so it does nothing (any command*
*. //that occurs while the string queue is*
DECODE:[String number one](1) *// this command puts the string into the string queue*
ADD:[2,g](1) *// this affects "String number one"*
DECODE:[Second String](2) *// this puts the string at the end of the string queue*
REMOVE:[g](2) *//this affects "String number one"*
SWAP:[e,S](2) *//this affects "Second string"*
REPLACE:[n,5](3) //this affects "String number one"
PRINT:(3) //this prints whatever "Second string" now equals
REMOVE:[ ](4) //this affects "String number one"
PRINT:(5) //this prints whatever "String number one now equals
REMOVE:[4](9) //the string queue is empty so this does nothing

**Expected output:**

eScond etring
Stri55umbero5e