ICS Lab3

Yunfan Li

3200102555

Introduction

In this lab, we were tasked with implementing a deque data structure in LC-3 assembly language. The deque allows elements to be added and removed from both the front and back of the list, making it a more versatile data structure than the stack or queue. Our implementation supports the following operations:

- '+' : push a character onto the left side of the deque
- '-' : pop a character from the left side of the deque and print it
- '[' : push a character onto the right side of the deque
- ']' : pop a character from the right side of the deque and print it

Algorithm

Our implementation of the deque uses two pointers, one for the left end and one for the right end of the deque. Initially, both pointers are set to the same memory location, indicating an empty deque. When a character is pushed onto the left side of the deque, the left pointer is decremented and the character is stored at that memory location. When a character is popped from the left side of the deque, the character at the left pointer is printed and the left pointer is incremented. The same process is used for pushing and popping from the right side of the deque, using the right pointer instead.

```
leftPointer = 0
rightPointer = 0

pushLeft(value):
    leftPointer = leftPointer - 1
    deque[leftPointer] = value

pushRight(value):
    deque[rightPointer] = value
    rightPointer = rightPointer + 1

popLeft():
    if leftPointer == rightPointer:
        print '_'
    else:
        print deque[leftPointer]
        leftPointer = leftPointer + 1

popRight():
    if leftPointer == rightPointer:
        print '_'
    else:
        rightPointer = rightPointer - 1
```

```
        print deque[rightPointer]

while true:
    inputChar = getInput()
    if inputChar == '+':
        value = getNextInputChar()
        pushLeft(value)
    else if inputChar == '-':
        popLeft()
    else if inputChar == '[':
        value = getNextInputChar()
        pushRight(value)
    else if inputChar == ']':
        popRight()
    else:
        // ignore unsupported input and prompt for input again
        continue
```

Explaination of Code

The code begins with initializing all registers to 0. Then, the three pointers are set to their appropriate initial memory locations. The program then enters an infinite loop, continuously prompting the user for input and performing the appropriate operation based on that input.

When the user enters '+' or '[', the program gets the next character from input and stores it at the appropriate end of the deque, incrementing or decrementing the appropriate pointer accordingly. When the user enters '-' or ']', the program checks if the deque is empty (i.e. the left and right pointers are equal), and if so, prints an underscore character. Otherwise, it prints the character at the appropriate end of the deque and increments or decrements the appropriate pointer.

The code includes two helper methods, SUCCESS and FAIL, which are used to restore the values of the registers after the program has executed a push or pop operation.

TA Question

Q: What if the deque allows Wrap-Around?

A: We need to add two parts in the code:

1. When touching the boundary after pushing, move the corresponding pointer to the other side.
2. When touching the boundary after popping, check the locations of the two pointers, then move the pointer back.