

Homework #3

(Introduction to Data Structures)

Due date: May 03, 2018

학번: 2014310407

이름: 이 준 혁(JunHyuk Lee)

1. Palindrome Checker

1.1. Solution

앞에서부터 리스트를 읽을 변수와 뒤에서부터 리스트를 읽을 변수를 선언해서, 각각 맨 앞과 맨 마지막의 문자를 읽는다. 만약 그 값이 알파벳 혹은 숫자가 아니라면, 그 문자를 지운 후에 다시 입력을 받는다.

앞과 뒤 모두 문자 혹은 숫자 입력을 받은 경우에는, 대문자인 경우에 소문자로 취급하기 위해, tolower 함수를 이용해 소문자로 모두 바꿨다. 이렇게 해서 비교한 값이 한 번이라도 다르면, palindrome 이 아니므로 바로 false 를 반환했다. 이 과정을 리스트의 원소가 하나 이하로 남을 때까지 반복했고, 이 과정까지 왔다면 palindrome 인 것이므로 true 를 반환했다.

1.2. Result (snapshot)



2. Swap Adjacent Two Blocks

2.1. Solution

총 길이 / (블록 길이 * 2)의 정수 부분만큼만 블록을 바꾸므로, 그 값을 계산해서 몇 번 반복할지를 결정했다.

그리고 세 개의 노드를 선언하고, 각각 첫 번째 블록의 맨 앞의 앞 노드, 두 번째 블록의 맨 앞의 앞 노드, 두 번째 블록의 마지막 노드를 가리키게 하고, 각각 node1,2,3 이라 이름을 붙였다.

그리고 첫 번째 블록의 맨 앞 노드를 임시로 저장한 후, node1 이 node2 의 next 를, node2 가 node3 의 next 를, node3 가 임시로 저장한 맨 앞의 노드를 가리키게 했다. 이렇게 하면 두 블록의 순서가 바뀌게 된다.

이 때 임시로 노드를 저장한 이유는, node1 이 가리키는 값이 바뀌기 때문에, node3 가 가리킬 곳을 확실하게 지정해 주기 위해서 였다. 이후 node1 을 node2(두 번째 블록의 맨 마지막 노드)로 바꾸어, 다시 변경을 진행했다. 이 과정을 이전에 계산한 총 반복 횟수만큼 반복했다.

2.2. Result (snapshot)

```
C:\WINDOWS\system32\cmd.exe
11
90 3 23 23 5 88 16 91 83 24 88
2
23 23 90 3 16 91 5 88 83 24 88 계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
6
92 81 5 4 23 7
3
4 23 7 92 81 5 계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
10
7 99 1 2 3 5 88 23 1 4
1
99 7 2 1 5 3 23 88 4 1 계속하려면 아무 키나 누르십시오 . . .
```

3. Discarding Cards

3.1. Solution

우선 리스트의 길이가 1 인 경우에는, 맨 앞의 노드의 값을 출력 후 종료했다.

리스트의 길이가 2 이상인 경우에는, 리스트의 길이가 2 이하가 될 때까지 다음과 같이 반복했다.
가장 먼저 맨 앞의 노드를 출력하고 지운 후에, Front 를 맨 앞의 노드 앞의 노드를 가리키게,
middle 을 중간 노드의 앞 노드를 가리키게, rear 는 맨 마지막 노드를 가리키게 설정했다.

이 경우에, 리스트의 길이가 짝수일 때와 홀수일 때가 다르므로 다른 방식으로 계산했다.

1. 짝수인 경우

짝수인 경우에는 rear 가 front 의 next 를 가리키게 하고, front 가 middle 의 next 를 가리키게,
그리고 middle 은 NULL 을 가리키게 했다. 이렇게 되면 앞 블록과 뒤 블록의 자리가 바뀌게 된다.

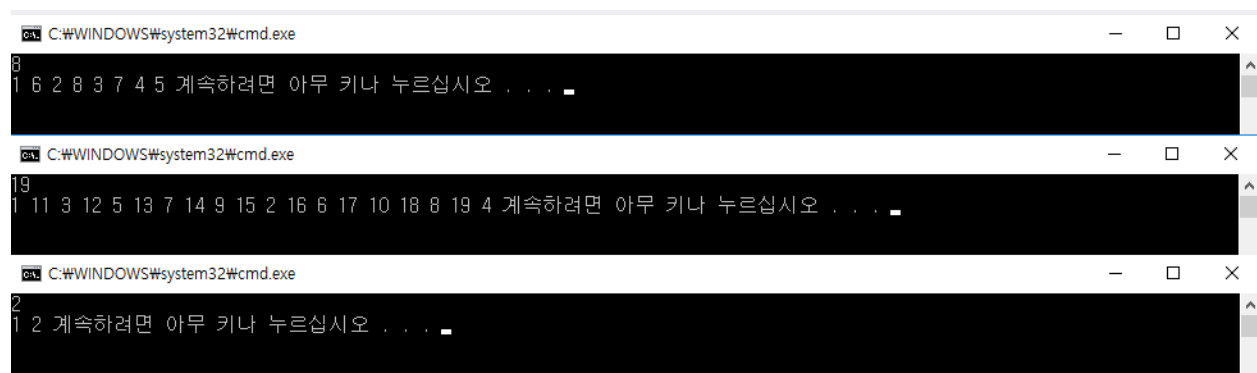
2. 홀수인 경우

홀수인 경우, 가운데의 블록 하나가 그 위치를 유지하므로, first 가 front 의 next 를 가리키게 한 후
다음과 같이 바꿨다.

Rear 가 middle 의 next 를(즉 가운데 블록), front 가 middle 의 next 의 next(가운데 노드를 건너뛰기 위함)를, middle 의 next 는 first 를 가리키도록 했다. 그 후에 middle 이 NULL 을 가리키게 하면서, 가운데 노드 하나를 제외한 양 쪽 블록의 위치를 변경했다.

이렇게 두 개 이하로 노드가 남게 되면, 맨 앞의 노드를 출력 후, 지우고, 마지막 노드를 출력했다.

3.2. Result (snapshot)



```
C:\WINDOWS\system32\cmd.exe
8
1 6 2 8 3 7 4 5 계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
19
1 11 3 12 5 13 7 14 9 15 2 16 6 17 10 18 8 19 4 계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
2
1 2 계속하려면 아무 키나 누르십시오 . . .
```

4. Key Logger

4.1. Solution

문자열을 저장할 리스트와 현재 커서의 위치를 알려줄 변수를 선언했다. 이 때 커서의 최초 위치는 아무것도 가리키고 있지 않은 상태이므로 -1 로 초기화했다.

그리고 입력을 다음과 같이 다섯 가지로 분류했다.

1. 알파벳 혹은 숫자 / 2. '<' / 3. '>' / 4. '+' / 5. '-'

1. 알파벳 혹은 숫자인 경우, 위치를 하나 증가시켜 커서를 이동시킨 후 커서 위치에 해당 문자를 삽입했다.

2. '<' 인 경우, 위치가 -1 보다 큰 경우 위치를 1 감소시켰다. 아닌 경우에는 무시했다.

3. '>'인 경우, 위치가 길이 - 1 보다 작은 경우 위치를 1 증가시켰다. 아닌 경우에는 무시했다.
4. '+'인 경우, 위치가 -1 인 경우에는 가리키는 값이 없으므로 무시했고, 아닌 경우에는 대문자일 때는 소문자로, 소문자일때는 대문자로 값을 변경했다.
5. '-'인 경우, 위치가 -1 인 경우에는 역시 가리키는 값이 없으므로 무시했고, 아닌 경우에는 위치에 해당하는 값을 지운 후에 위치를 1 감소시켰다.

이렇게 주어진 입력을 처음부터 하나씩 받아 지워 나가면서, 입력이 모두 없어질 때까지 반복했다. 이후 저장된 문자열 리스트를 출력했다.

4.2. Result (snapshot)

```

C:\WINDOWS\system32\cmd.exe
8
A B E F D G 3 1
A B E F D G 3 1 계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
10
< a + B C - 8 < k +
A B K B 계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
15
- - > < a + + 7 + F C - d +
a 7 F D 계속하려면 아무 키나 누르십시오 . . .

```

5. Calculating Two Numbers

5.1. Solution

우선 입력 받은 숫자들을 뒤에서부터 읽어 나가며 더하거나 빼려고 했다. 이 과정에서 두 개의 합이 10 이 넘는 경우나 어떤 자릿수에서 차이가 음수가 되는 경우가 발생했기 때문에, 이를 관리하기 위해 digit 이라는 변수를 선언했다.

우선 +인 경우와 -인 경우로 나눴다.

더하기인 경우에는 두 숫자를 뒤에서부터 읽는데, 숫자 리스트가 비어 있지 않다면 그 값을 저장한 후 그 값을 리스트에서 지웠다. 만약 비어 있었다면 0으로 값을 초기화했다. 이렇게 얻은 두 값을 더하는데, 10 보다 크면 10 을 빼서 결과 리스트에 넣고, digit 에 1 을 저장해서 한 자릿수 위에서 1 을 더할 수 있도록 했다. 아닌 경우에는 그대로 리스트에 넣고 digit 을 0 으로 초기화했다. 이 때 아랫자리에서 10 을 초과했을 수도 있으므로, 값을 리스트에 넣을 때 digit 만큼을 더해 아랫자리의 계산 값이 반영될 수 있도록 했다.

뺄 경우에도 읽는 행위는 더하는 것과 같이 하되, 0 보다 작으면 10 을 더해서 결과 리스트에 넣고, digit 에 -1 을 저장해서 한 자릿수 위에서 1 을 뺄 수 있도록 했다. 아닌 경우에는 그대로 리스트에 넣고 digit 을 0 으로 초기화했다. 이 때 아랫자리에서의 계산한 값이 0 보다 작았을 수 있으므로, 값을 리스트에 넣을 때 digit 만큼을 더해 아랫자리의 계산 값이 반영될 수 있도록 했다.

이 과정을 두 리스트가 전부 빌 때까지 반복했다. 다만 더하기의 경우에는 두 숫자의 최대 자릿수의 합이 10 보다 커져 원래 숫자보다 길이가 길어질 수 있으므로, 마지막 digit 이 1 인 경우에는 1 을 추가로 결과 리스트에 넣었다.

이 과정이 모두 끝난 후, 마지막으로 leading zero 들을 다 제거했는데, 다만 0 그 자체인 경우에는 제거하면 안 되므로 길이가 1 보다 클 때의 leading zero 들만 제거했다.

5.2. Result (snapshot)

