

알고리즘 실습 보고서

-Knapsack&Sequence-

전공 :컴퓨터공학과

분반 : 05반

학번 :201701988

이름 :김수빈

1. 실행 환경

본 실습은 Windows10 64bit, jdk 1.8.0_221, Eclipse EE가 설치된 환경에서 실행되었다

2. 과제 설명

- 출제된 과제에 대한 설명

이번 과제 Knapsack&Sequence 는 data09_knapsack.txt 파일을 읽어 차례대로 Item들의 number, value, weight를 읽고 배낭의 무게제한을 넘지 않으면서 최대한의 value를 가질 수 있는 Item 조합을 구하는 것이다.

3. 문제 해결 방법

- 문제를 해결하기 위해 자신이 사용한 방법, 아이디어에 대한 설명

먼저, Item의 number, value, weight를 표현하기 위해, int형 값 number와 value, weight를 갖는 Item 객체와, 또 $OPT(i,w)$ 에서의 Item 조합, $OPT(i,w)$ 에서의 전체 value값을 표현하기 위해 OPT 객체를 생성했다.

주어진 문제를 해결하기 위해 필요한 메소드는 OPT 테이블을 채우는 메소드 getOPT와 또 입력받은 무게 제한 안에서 최대 value값을 갖는 Item 조합을 출력하는 메소드 findMaxValue가 있다.

data09_knapsack.txt 파일을 읽어 ArrayList<Item>을 구성한다. ArrayList<Item> itemList와 item 개수 n과 입력받은 무게제한값 w를 매개변수로 하여 getOPT 메소드를 호출한다.

#OPT Table 채우기

getOPT 메소드에서는 아래와 같이 무게 제한에 따른 Item 조합에서 최대 value값과 그때의 Item 조합을 저장하는 OPT 값을 배열로 저장하는 OPT_Table을 사용하여 해결한다.

//그림//

i (0 ≤ i ≤ n)와 j(0 ≤ j ≤ w)에 다음을 반복한다.

1. 0일 경우, 무게제한에 상관없이 item들이 없는 것으로, 모두 0을 total_value로 갖고 빈 ArrayList를 item조합으로 하는 OPT를 생성하여 채운다.

2. itemList의 첫 번째부터 i번째 Item까지 무게 제한이 j일 때의 최대 value를 구한다. 1번째 Item의 무게가 무게 제한을 넘을 경우, 배낭에 삽입하지 못하므로 $OPT[i][j]$ 는 $OPT[i-1][j]$ 와 같다. $OPT[i-1][j]$ (itemList의 i-1까지 Item들 중에서 j의 무게제한을 가질때의 value가, 1-1까지 Item들 중에서 (j- i번째 Item의 무게)의 무게제한을 가질 때의 value와 1번째 Item의 value를 더한 값보다 더 클 경우, $OPT[i][j]$ 는 $OPT[i-1][j]$ 와 같다. 아니면 itemList의 1-1까지 Item들 중에서 (j- i번째 Item의 무게)의 무게제한을 가질때의 Item조합을 복사하여, 거기에 1번째 Item을 삽입하여 $OPT[i][j]$ 의 item 조합으로 하고, value를 1-1번째 Item까지 j-1번째 item 무게의 무게 제한을 가질때의 value를 $OPT[i][j]$ 의 total_value로 삼는다.

Item 조합 구하기

배열(Table)에 OPT 값을 채우는 함수와, 완성된 배열을 분석하여 가치(value) 총합이 가장 높은 item 구성 및 value 합을 출력하는 함수를 각각 구현하기 위해, 출력하는 함수 findMaxValue를 따로 구현하였다. OPT배열을 매개변수로 받아 OPT 테이블의 total_value를 모두 출력한 후, 테이블의 우하단의 OPT 객체의 total_value와 item 조합을 출력한다.

homework2

	0	t	c	a	t	a	g	t	t	a	a	c	a
0	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
t 1	-2	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21
c 2	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
a 3	-6	-3	0	3	1	-1	-3	-5	-7	-9	-11	-13	-15
g 4	-8	-5	-2	1	2	0	0	-2	-4	-6	-8	-10	-12
a 5	-10	-7	-4	-1	0	3	1	-1	-3	-3	-5	-7	-9
a 6	-12	-9	-6	-3	-2	1	2	0	-2	-2	-2	-4	-6
g 7	-14	-11	-8	-5	-4	-1	2	1	-1	-3	-3	-3	-5
t 8	-16	-13	-10	-7	-4	-3	0	3	2	0	-2	-4	-4
a 9	-18	-15	-12	-9	-6	-3	-2	1	2	3	1	-1	-3
c 10	-20	-17	-14	-11	-8	-5	-4	-1	0	1	2	2	0
c 11	-22	-19	-16	-13	-10	-7	-6	-3	-2	-1	0	3	1

t	c	a	t	a	g	t	t	a	a	c	a
t	c	a	g	a	a	g	t	-	a	c	c

	0	t	c	a	t	a	g	t	t	a	a	c	a
0	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
t 1	-2	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21
c 2	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
a 3	-6	-3	0	3	1	-1	-3	-5	-7	-9	-11	-13	-15
g 4	-8	-5	-2	1	2	0	0	-2	-4	-6	-8	-10	-12
a 5	-10	-7	-4	-1	0	3	1	-1	-3	-3	-5	-7	-9
a 6	-12	-9	-6	-3	-2	1	2	0	-2	-2	-2	-4	-6
g 7	-14	-11	-8	-5	-4	-1	2	1	-1	-3	-3	-3	-5
t 8	-16	-13	-10	-7	-4	-3	0	3	2	0	-2	-4	-4
a 9	-18	-15	-12	-9	-6	-3	-2	1	2	3	1	-1	-3
c 10	-20	-17	-14	-11	-8	-5	-4	-1	0	1	2	2	0
c 11	-22	-19	-16	-13	-10	-7	-6	-3	-2	-1	0	3	1

t	c	a	t	a	g	t	t	a	a	c	a
t	c	a	g	a	a	g	t	a	-	c	c

		0	t	c	a	t	a	g	t	t	a	a	c	a
	0	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
t	1	-2	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21
c	2	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
a	3	-6	-3	0	3	1	-1	-3	-5	-7	-9	-11	-13	-15
g	4	-8	-5	-2	1	2	0	0	-2	-4	-6	-8	-10	-12
a	5	-10	-7	-4	-1	0	3	1	-1	-3	-3	-5	-7	-9
a	6	-12	-9	-6	-3	-2	1	2	0	-2	-2	-2	-4	-6
g	7	-14	-11	-8	-5	-4	-1	2	1	-1	-3	-3	-3	-5
t	8	-16	-13	-10	-7	-4	-3	0	3	2	0	-2	-4	-4
a	9	-18	-15	-12	-9	-6	-3	-2	1	2	3	1	-1	-3
c	10	-20	-17	-14	-11	-8	-5	-4	-1	0	1	2	2	0
c	11	-22	-19	-16	-13	-10	-7	-6	-3	-2	-1	0	3	1

t	c	a	t	a	g	t	t	a	a	c	a
t	c	a	g	a	a	g	t	a	c	c	-

x: tcataattaca / y: tcgaatacc

서열	X	0	1	2	3	4	5	6	7	8	9	10	11	12
y	0	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
1	1	-2	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21
2	2	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
3	3	-6	-3	0	3	1	-1	-3	-5	-7	-9	-11	-13	-15
4	4	-8	-5	-2	1	2	0	0	-2	-4	-6	-8	-10	-12
5	5	-10	-7	-4	-1	0	3	1	-1	-3	-3	-5	-7	-9
6	6	-12	-9	-6	-3	-2	1	2	0	-2	-2	-2	-4	-6
7	7	-14	-11	-8	-5	-4	-1	2	1	-1	-3	-3	-3	-5
8	8	-16	-13	-10	-7	-4	-3	0	3	2	0	-2	-4	-4
9	9	-18	-15	-12	-9	-6	-3	-2	1	2	3	1	-1	-3
10	10	-20	-17	-14	-11	-8	-5	-4	-1	0	1	2	2	0
11	11	-22	-19	-16	-13	-10	-7	-6	-3	-2	-1	0	3	1

결과 화면

```
Console
<terminated> knapsack_problem [Java Application] C:\Program Files\Ja
배낭의 사이즈를 입력하세요. (0~50) : 11
| 0 0 0 0 0 0 0 0 0 0 0 0
| 0 1 1 1 1 1 1 1 1 1 1 1
| 0 1 6 7 7 7 7 7 7 7 7 7
| 0 1 6 7 7 18 19 24 25 25 25 25
| 0 1 6 7 7 18 22 24 28 29 29 40
| 0 1 6 7 7 18 22 28 29 34 35 40
max : 40
item : 3 4
```

5. 느낀점 및 고찰

homework2의 문자열 비교는 문자열이 너무 길어 계산이 헛갈렸습니다. 하지만 큰 어려움 없이 해결할 수 있었습니다.