2020년 가을학기 알고리즘

Dynamic Programming

데이터네트워크연구실
문현수, 이영석
munhyunsu@cs-cnu.org

12주차 Feedback

- 제출률: 69%

이번주 실습 목표

- Dynamic Programming
  - 동적 프로그래밍 활용
  - Top-down / Bottom-up DP
  - DP 대표 문제

- 실습1: Fibonacci (1점)
- 실습2: Knapsack (3점)
- 실습3: LumberMill (3점)
- 실습4: FrogGame (3점)

실습1. Fibonacci (1)

피보나치 수열

- 0, 1로 시작해 이전 2개 수를 더한 값의 수열 (https://en.wikipedia.org/wiki/Fibonacci_number)

Sequence properties [edit]

The first 21 Fibonacci numbers F_n for n = 0, 1, 2, ..., 20 are:[49]

F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
0	1	1	2	3	5	8	13	21	34	55	89	144	233	377	610	987	1597	2584	4181	6765

The sequence can also be extended to negative index n using the re-arranged recurrence relation

$$F_{n-2} = F_n - F_{n-1},$$

which yields the sequence of "negafibonacci" numbers[50] satisfying

$$F_{-n} = (-1)^{n+1} F_n.$$

Thus the bidirectional sequence is

F_{-8}	F_{-7}	F_{-6}	F_{-5}	F_{-4}	F_{-3}	F_{-2}	F_{-1}	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
−21	13	−8	5	−3	2	−1	1	0	1	1	2	3	5	8	13	21

Fibonacci 로 이해하는 Dynamic Programming

- 2가지 조건을 만족할 때 DP라고 부름
  a. Optimal substructure: 작은 최적 답의 조합으로 답을 찾을 수 있음
  b. Overlapping subproblem: 작은 문제가 반복해서 나옴
- 2가지 접근법이 있음
  a. Top-down: 큰 문제를 작은 문제로 쪼개면서 해결. Memorize (Cache) 사용
  b. Bottom-up: 작은 문제의 조합으로 큰 문제를 해결.

T.D. vs. B.U.

```
function fib(n)
    if n <= 1 return n
    return fib(n - 1) + fib(n - 2)
```

```
var m := map(0 → 0, 1 → 1)
function fib(n)
    if key n is not in map m
        m[n] := fib(n - 1) + fib(n - 2)
    return m[n]
```

```
function fib(n)
    if n = 0
        return 0
    else
        var previousFib := 0, currentFib := 1
        repeat n - 1 times // loop is skipped if n = 1
            var newFib := previousFib + currentFib
            previousFib := currentFib
            currentFib  := newFib
    return currentFib
```

Big Fibonacci

- 피보나치 수열의 N번째 항을 출력하시오.
  - JAVA 참고) https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/math/BigInteger.html

[Input]
0

[Output]
0

[Input]
5

[Output]
5

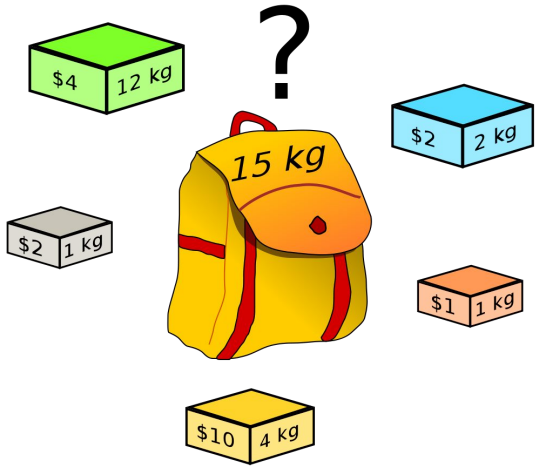
[Input]
20

[Output]
6765

실습2. Knapsack (3)

배낭 문제: 조합형 문제

- Dynamic Programming 연습 문제 대표
- 각 물건은 무게와 가치가 있음
- 배낭에는 허용 최대 무게가 있음
- 배낭에 넣을 수 있는 최대 가치는?



해결 방법

- 무차별 접근
  a. 모든 물건을 조합 (0-1)
  b. 가방에 들어가는지 확인
  c. 가방에 들어가면 가치 비교 후 저장

- 동적 계획법
  a. 아무것도 안 들었을 때 가치 0
  b. 물건을 하나 선택
  c. 가방 무게를 0부터 늘려가며 가치 비교
     - 선택한 물건을 안 넣었을 때
     - 선택한 물건을 넣었을 때

0-1 Knapsack 문제

- 가방에 물건을 넣어 옮기려한다. 배낭에는 넣을 수 있는 물건 크기에 한계가 있으며, 각 물건은 크기와 가치가 부여되어있다. 배낭에 넣을 수 있는 물건들의 최대 가치를 구하시오.
    - 입력)
      배낭크기
      물건 개수
      물건 크기 리스트
      물건 가치 리스트

```
[Input]
5
4
2 3 4 5
3 4 5 6

[Output]
7
```

실습3. Lumber Mill (3)

제재소

- 재목을 만드는 제재소에는 재료로 들어온 나무를 정확히 K미터로 자르는 기계가 있다.
  재료로 들어온 나무 관리를 위하여, 나무를 잘라 가능한 적은 갯수로 보관하려 한다.
  예를 들어, 1 3 7미터로 자르는 기계가 있고 15미터 재료가 들어오면 7, 7, 1미터로 자르는 것이 가장 적은 나무 조각으로 자르는 방법이다.
  나무 자르는 기계의 크기와, 재료 나무가 주어졌을 때 최소 나무 조각을 구하시오.

```
[Input]
3
3 7 1
15

[Output]
3

[설명]
1 * 15 >
3 * 5  >
1, 7, 7
```

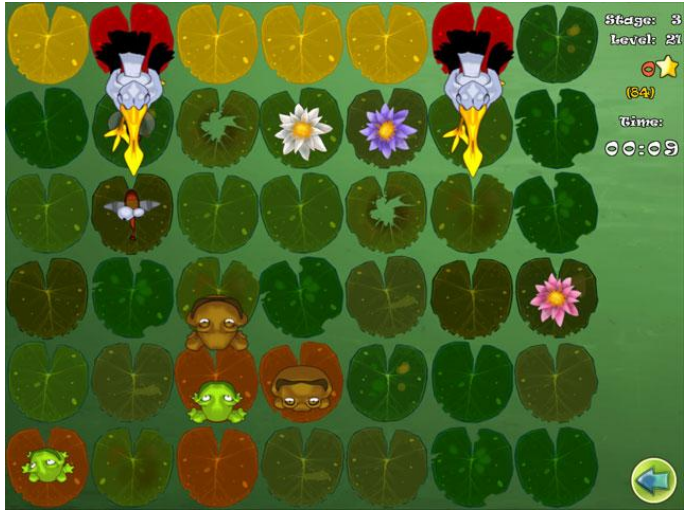
실습4. Frog Game (3)

개구리 게임

- 개구리 게임은 화면 아래에서 화면 위로 발판을 밟아가며 올라가는 게임이다.
  발판에 도착하면 각 발판에 배정되어있는 점수를 획득하며, 개구리는 왼쪽위, 위, 오른쪽위 발판으로 이동할 수 있다.
  개구리 게임 발판 점수가 주어졌을 때, 개구리 게임에서 얻을 수 있는 최대 점수를 계산하시오.
  - 입력)
    행 렬
    점수리스트 (열)

[Input]
4 4
1 2 5 0
3 2 0 6
1 4 2 1
5 1 3 2

[Output]
16



참고

Standard Library

- Python 3: https://docs.python.org/3/library/
- JAVA 11: https://cr.openjdk.java.net/~iris/se/11/latestSpec/api/
- C: https://en.cppreference.com/w/c
- CPP: https://en.cppreference.com/w/cpp

ID/ext	external ID	name
adb	ada	Ada
awk	awk	AWK
bash	bash	Bash shell
c	c	C
csharp	csharp	C#
cpp	cpp	C++
f95	f95	Fortran
hs	haskell	Haskell
java	java	Java
js	javascript	JavaScript
kt	kotlin	Kotlin
lua	lua	Lua
pas	pascal	Pascal
pl	pl	Perl
sh	sh	POSIX shell
plg	prolog	Prolog
py2	python2	Python 2
py3	python3	Python 3
r	r	R
rb	ruby	Ruby
scala	scala	Scala
swift	swift	Swift

주요 링크

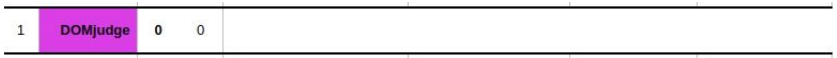
- 코딩테스트: http://coding.cnu.ac.kr:8080/domjudge/public
- FAQ : https://docs.google.com/document/d/1ntR6GS1SI7dRbYlw-pu8uT8U65Wc-RtMj10IEpiapfU/
- 질문: 메일!
- 문의 사항: munhyunsu@cs-cnu.org
  - ID / PASSWORD 변경
  - 실습 시간외 질문: [AL]질문제목

잊지 않아야 할 것) 소스코드 및 보고서

- 사이버 캠퍼스에 목요일까지 제출
  - 추가 시간 필요한 학생들도 목요일까지 제출. 추가 시간 문제 해결은 메일로도 제출!
- 보고서(.pdf 파일), 소스코드(.java, .py 등) zip 파일 압축
  - AL_학번_이름_06.zip (메일 추가 제출) or AL19_06.zip (사이버캠퍼스)
- 시간/공간 복잡도 해석(STL 고려), 자신의 생각, 질문, 느낀점, 공유하고 싶은 문제
  - 문제 해결을 위해 어떤 접근법을 사용하였는지, 무엇을 배웠고 느꼈는가?

보고서 템플릿: .pdf 로 제출!

- 알고리즘-x주차-주제
  학번 이름
- 코드 테스팅 결과 (점수표)



- 각 문제별 내용
  a. 문제 / 목표
  b. 해결 방법 (주요 소스코드 첨부. 스크린샷 or 코드 CV)
  c. 결과 (입력, 출력 결과)
- 느낀점: 과제를 하며 느낀 점 / 공유하고싶은 문제 / 난이도 / 부탁 / 조교에게... 등