Univerzitet u Tuzli

Fakultet Elektrotehnike

Uvod u Računarske Algoritme

Zadaća 3

Dinamičko programiranje, backtracking rekurzija

Tuzla, 21. 5. 2020.

Napomena

U svim problemima koji slijede nije dozvoljena upotreba komandi i funkcija koje dosad nisu korištene na predavanjima ili vježbama. Smije se koristiti std::sort ukoliko je potrebno. Dozvoljena je upotreba kontejnera iz standardne biblioteke (std::vector i std::list), kao i C nizova. Nerekurzivna rješenja se ne mogu smatrati tačnim ukoliko je u zadatku naglašeno da je potrebno koristiti rekurziju.

Zadatak 1.

Neka je sekvenca cijelih brojeva definisana na sljedeći način:

```
a(0) = 2
a(1) = 4
a(2) = 12
a(n) = 3*a(n-3)+2*a(n-2)+a(n-3)
```

Potrebno je napisati funkciju koja će za proslijeđeni cijeli broj n vratit n-ti broj u prethodno definisanoj sekvenci. Rješenje implementirati rekurzivno bez memoizacije, sa memoizacijom, te nerekurzivno primjenom principa dinamičkog programiranja.

Zadatak 2.

Potrebno je napisati funkciju koja će za proslijeđeni cijeli broj n vratiti broj različitih načina na koje možemo dobiti broj n kao zbir 1, 3 i 4. Rješenje implementirati rekurzivno bez memoizacije, sa memoizacijom, te nerekurzivno primjenom principa dinamičkog programiranja. Naprimjer, za n = 5 imamo 6 različitih kombinacija brojeva 1, 3 i 4 čijim sabiranjem možemo dobiti 5: 5 = 1 + 1 + 1 + 1 + 1 = 1 + 1 + 3 = 1 + 4

Napomena: Kombinacije (1 + 1 + 3), (1 + 3 + 1) i (3 + 1 + 1) smatramo istom kombinacijom i brojimo je samo jednom, jer je riječ o istim brojevima samo u drugom poretku.

Zadatak 3 - Knapsack problem

Dječak je došao u zabavni park sa ukupno 10KM i želi potrošiti što više vremena na različitim vožnjama. Pošto svaka vožnja ima različitu cijenu i dužinu trajanja, dječak je odlučio napraviti efikasan algoritam koji će mu u što kraćem roku dati optimalno rješenje sa stanovišta utrošenog vremena u skladu sa njegovim budžetom. Zadatak je implementirati takav algoritam uzimajući u obzir da dječak nije htio ni na jednu vožnju ići više od jednom.

Tip vožnje	cijena	trajanje (min)
1	10	40

Tip vožnje	cijena	trajanje (min)
2	5	18
3	4	35
4	2	2
5	3	4
6	1	10

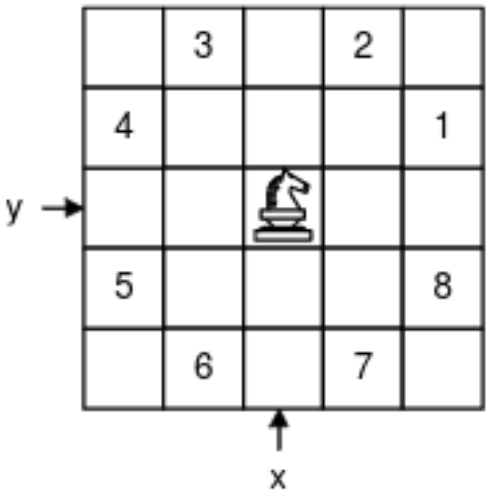
Algoritam treba da vrati ukupno trajanje svih odabranih vožnji. Također, za drugi dio zadatka, potrebno je osmisliti i implementirati funkcionalnost za ispis tipova vožnji koje je najbolje odabrati da bi ukupno trajanje bilo maksimalno.

Zadatak 4 - kockice

Kocka se baca n puta. Realizovati algoritam koji ispisuje sve kombinacije rezultata tih bacanja za koje je ukupna suma manja od S. n i s su parametri pri pozivu fukncije.

Zadatak 5 - konjićev skok

Konj ili skakač je postavljen na šahovsko polje sa koordinatama 0,0. Kreće se prema uobičajnim pravilima šaha skačući na neko od polja koje se nalazi u obliku slova L od trenutne pozicije (slika).



Potrebno je pronaći niz skokova koji će obići svih 64 polja šahovske ploče, pri čemu se na svako polje može skočiti samo jednom. Nakon što se naiđe na prvi

ovakav obilazak, algoritam staje sa radom, a korisniku se ispisuje šahovska ploča sa koracima. Primjer jednog obilaska je dat ispod:

```
 1 38 59 36 43 48 57 52
60 35  2 49 58 51 44 47
39 32 37 42  3 46 53 56
34 61 40 27 50 55  4 45
31 10 33 62 41 26 23 54
18 63 28 11 24 21 14  5
 9 30 19 16  7 12 25 22
64 17  8 29 20 15  6 13
```