Zadaća 3 iz predmeta Uvod u Računarske Algoritme

Zadatak 1. - Jednostavna rekurzija

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,
- nerekurzivno primjenom principa dinamičkog programiranja.

Zadatak 2 - Collatz sekvenca

Collatzova sekvenca je rekurzivna sekvenca koja glasi:

```
a(n+1) = a(n) / 2, ako je a(n) parno,
a(n+1) = 3 * a(n) + 1, ako je a(n) neparno
```

pretpostavka je da se za svaki prirodni broj ova sekvenca eventualno svede na broj 1, primjera radi, za broj 3 imamo sekvencu:

3 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1

Napisati funkciju `int collatz(unsigned long n)` koja vraća natrag broj koraka koji je potreban kako bi se inicijalni broj sveo na 1. Za primjer broja 3, to bi bilo 7 koraka.

Odrediti koji broj u intervalu [2, 100] ima najdužu putanju do broja 1?

Pokušati optimizirati rješenje koristeći memoizaciju.

> Obratite pažnju da neke jednostavne početne vrijednosti Collatz sekvence mogu dovesti do jako visokih brojeva kroz samu sekvencu, pa bi bilo dobro napraviti neki vid kombinovanog rješenja. Primjera radi, prvih 1000 vrijednosti bi mogli spremiti u lookup tabelu, a vrijednosti preko toga računati regularnim pristupom rekurzije bez tabele.

Zadatak 3. - Najduža rastuća podsekvenca

Napisati funkciju koja pronalazi dužinu najduže rastuće podsekvence. Na primjeru vidimo, ako je dat niz

10, 22, 9, 33, 21, 50, 41, 60, 80

Najduža rastuća podsekvenca bi bila

10, 22, 33, 50, 60, 80

i njena dužina je 6.

https://www.geeksforgeeks.org/longest-increasing-subsequence-dp-3

Riješiti problem koristeći dinamičko programiranje.

Ako je dat file ulaz1.txt

```
10 22 9 33 21 50 41 60 80
```

Nakon pokretanja programa sa datom komandom, izlaz će biti:

```
$ ./a.out < ulaz1.txt
6
```

Zadatak 4. - Maksimalna podsuma niza

Napisati program koji će pronaći najveću sumu unutar niza, uzimajući proizvoljan broj uzastopnih elemenata tog niza.

Ako je dat file ulaz1.txt

```
-2 -3 4 -1 -2 1 5 -3
```

Nakon pokretanja programa sa datom komandom, izlaz će biti:

```
$ ./a.out < ulaz1.txt
4 -1 -2 1 5
7
```

Upravo uzimanjem uzastopnih vrijednosti 4, -1, -2, 1 i 5 dobijemo najveću sumu, koja iznosti 7, i to su vrijednosti na izlazu iz programa.

Riješiti problem koristeći dinamičko programiranje.

https://www.geeksforgeeks.org/largest-sum-contiguous-subarray

Zadatak 5. - Maksimalna podsekvenca palindrom

Napisati prgram koji će pronaći najdužu podsekvencu unutar niza koja je ujedno palindrom.

Na primjer, ako bi imali niz GEEKSFORGEEKS, najduža podsekvenca bi bila EEKEE i njena dužina iznosi 5.

https://www.geeksforgeeks.org/longest-palindromic-subsequence-dp-12

Backtracking

Zadatak 6 - kockice

Kocka se baca `n` puta. Realizovati algoritam koji ispisuje sve kombinacije rezultata tih bacanja za koje je ukupna suma manja od S.

Zadatak 7 - Labirint

Dat je labirint, predstavljen kao matrica vrijednosti 0 i 1. 0 simbolizira da se tim poljem labirinta može proći, dok 1 predstavlja prepreku za prolazak. Hrčak polazi s koordinata 0,0 i kreće se prema koordinatama N-1, M-1. Može da se kreće samo u dva pravca, naprijed i dolje.

Odrediti najkraću putanju kojom hrčak može da stigne do destinacije i ispisati je korisniku na ekran.

Ako je dat file ulaz1.txt

```
4 4
1 0 0 0
1 1 0 1
0 1 0 0
1 1 1 1
```

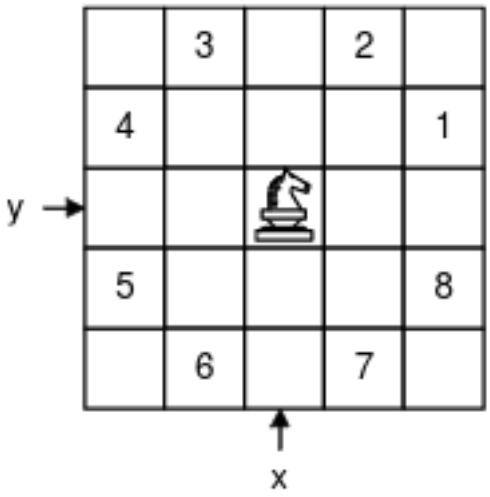
Gdje su prve dvije vrijednosti dimenzije matrice, N i M, a zatim je dat labirint. Nakon pokretanja programa sa datom komandom, izlaz će biti:

```
$ ./a.out < ulaz1.txt
1 0 0 0
1 1 0 0
0 1 0 0
0 1 1 1
```

https://www.geeksforgeeks.org/rat-in-a-maze-backtracking-2

Zadatak 8 - 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:

```
$ ./a.out
  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
```

https://www.geeksforgeeks.org/the-knights-tour-problem-backtracking-1

Zadatak 9 - Sudoku

Sudoku predstavlja oblik slagalice gdje se u matricu dimenzija 9x9 unose brojevi od 1 do 9, na način da u svakom redu, svakoj koloni, i svakoj oblasti od 3x3, svaki broj nalazi samo jednom. Potrebno je razviti algoritam koji će za datu slagalicu sa ulaza, popuniti brojeve od 1 do 9 i naći rješenje.

Ako je dat file ulaz1.txt

```
3 0 6 5 0 8 4 0 0
5 2 0 0 0 0 0 0 0
0 8 7 0 0 0 0 3 1
```

```
0 0 3 0 1 0 0 8 0
9 0 0 8 6 3 0 0 5
0 5 0 0 9 0 6 0 0
1 3 0 0 0 0 2 5 0
0 0 0 0 0 0 0 7 4
0 0 5 2 0 6 3 0 0
```

Nakon pokretanja programa sa datom komandom, izlaz će biti:

```
$ ./a.out < ulaz1.txt
3 1 6 5 7 8 4 9 2
5 2 9 1 3 4 7 6 8
4 8 7 6 2 9 5 3 1
2 6 3 4 1 5 9 8 7
9 7 4 8 6 3 1 2 5
8 5 1 7 9 2 6 4 3
1 3 8 9 4 7 2 5 6
6 9 2 3 5 1 8 7 4
7 4 5 2 8 6 3 1 9
```

https://www.geeksforgeeks.org/sudoku-backtracking-7

Način predavanja

Zadaću je potrebno predati u obliku arhive *ime_prezime_zadaca2.zip*, pri čemu sadržaj arhive treba da bude direktorij *ime_prezime_zadaca2*, sa poddirektorijem za svaki zadatak, sa imenima: zadatak1, zadatak2, . . . , zadatak8.

Svaki zadatak je potrebno riješiti unutar jednog filea, sa nazivom *main.cpp*. Zadaci će biti testirani sa grupom ulaznih fileova, tako da je potrebno pridržavati se formata ulaza, koji je demonstriran za svaki zadatak pojedinačno, u vidu komande `./a.out < ulaz1.txt`.

Pošto je ostalo manje vremena predviđeno za izradu zadaće, u sklopu svakog zadatka se nalazi i link na stranicu sa objašnjenjem zadatka, koje je potrebno pročitati i razumjeti gradivo. Za sva dodatna pitanja se možete javiti putem maila.