LV05 - POLJA I VEKTORI KAO ARGUMENTI FUNKCIJE

Prijenos vektorske strukture funkciji

Modul	Programiranje
SIU	**Funkcije i rekurzija u programiranju**
Ishod učenja	**I3 - Upotrijebiti polje kao argument funkcije.**
Trajanje	2 školska sata - učenje temeljeno na radu
Način rada	Samostalno; pri dijeljenju računala izmjena uloga nakon prvog zadatka
Fokus	vector, 1D i 2D polje kao argumenti funkcije; izbor algoritma, rubni slučajevi i testiranje

Cilj vježbe

U problemima treba prepoznati što je potrebno pregledati, koje podatke funkcija treba primiti, koji rezultat vratiti i kako provjeriti rubne slučajeve.

- upotrijebiti vector kao argument funkcije i koristiti njegovu veličinu;
- primijeniti 1D ili 2D polje kao argument kada to odgovara strukturi problema;
- izvesti uvjet iz priče i pretvoriti ga u algoritam;
- odabrati test koji može otkriti pogrešku, a ne samo potvrditi očit primjer;
- objasniti zašto funkcija ne mora uvijek pregledati sve raspoložive podatke.

Prije kodiranja

Za svaki zadatak prije pisanja programa u komentar zapiši:
ULAZNI PARAMETRI -> ŠTO FUNKCIJA TRAŽI / RAČUNA -> ŠTO VRAĆA ILI MIJENJA.

Predložak komentara na početku .cpp datoteke

/* LV1 – Zadatak: _____

Ucenik/ucenici: _____

Funkcija prima: _____

Funkcija vraca: _____

Moj test: ulaz: _____

ocekivani rezultat: _____

sto test provjerava: _____

AI: nisam koristio / koristio za _____ Prompt (ako je koristen): _____ */

Dakle, napiši barem jedan svoj test i očekivani rezultat. Tek tada pokreni program.

AI možeš koristiti nakon vlastitog pokušaja kao pomoć pri objašnjenju pogreške ili kao 'protivnika' koji predlaže test za koji misli da bi mogao srušiti tvoj program. Ne traži gotovo rješenje; moraš moći objasniti predani kod.

1. PUNIONICE ZA E-BICIKLE

Na biciklističkoj ruti aplikacija prati zauzetost punionica. U vektoru je za svaku punionicu zapisan broj slobodnih priključaka. Grupa sudionika treba istodobno priključiti K bicikala na istoj punionici.

Napiši funkciju koja prima vektor brojeva slobodnih priključaka i broj K te vraća indeks prve punionice na kojoj ima mjesta za svih K bicikala. Ako takva punionica ne postoji, funkcija vraća -1.

Predloženi potpis:

```
int prvaPunionica(const vector<int>& slobodno, int K);
```

slobodno	K	rezultat
{1, 0, 4, 2, 5}	3	2
{2, 2, 2}	3	-1
{4, 1, 5}	4	0
{0, 0, 6}	6	2

Razmisli prije koda:

- Zašto funkciji ne treba poseban parametar n?
- Zašto je ovdje važno pronaći prvu odgovarajuću punionicu, a ne samo provjeriti postoji li neka?
- Može li funkcija završiti prije nego što pregleda cijeli vektor? Kada?

Tvoj test: _____ Očekivani rezultat: _____

2. SIGNAL U TUNELU

Robot prolazi kroz tunel i svakih nekoliko metara mjeri jačinu signala. Mjerenja su spremljena u 1D polje. Signal je stabilan dok se dva uzastopna mjerenja ne razlikuju za više od D jedinica.

Napiši funkciju koja vraća indeks prvog mjerenja na kojem je nastao prevelik skok u odnosu na prethodno mjerenje. Ako takvog skoka nema, vrati -1. Smatraj da polje ima barem dva elementa.

Predloženi potpis:

```
int prviSkok(int signal[], int n, int D);
```

signal	D	rezultat
{40, 42, 41, 49, 50}	5	3
{20, 24, 19, 23}	5	-1
{10, 16, 17}	5	1
{30, 25, 20}	5	-1

Važno: razlika može nastati u oba smjera. Pokušaj matematički izraziti udaljenost dvaju uzastopnih mjerenja tako da ne pišeš dva odvojena slučaja.

Odgovori u komentaru:

- Zašto petlja ne treba početi od elementa s indeksom 0?
- Zašto je zadnji test važan za razlikovanje > D od >= D?
- Koliko elemenata mora funkcija pregledati ako se prvi veliki skok pojavi vrlo rano?

Tvoj rubni test: _____ Očekivani rezultat: _____

3. AUTONOMNI ROVER - SIGURAN KORIDOR

Na neistraženom terenu planete rover istražuje područje predstavljeno pravokutnom mrežom visina. U svakom polju nalazi se visina terena. Rover se u ovom zadatku ne kreće proizvoljno: treba prijeći jednim od redaka slijeva nadesno.

Redak je SIGURAN ako razlika između najveće i najmanje visine u tom retku nije veća od dopuštene vrijednosti D. Rover želi odabrati siguran redak s najmanjom takvom razlikom. Ako ih je više jednako dobrih, bira redak s manjim indeksom. Ako nijedan redak nije siguran, rezultat je -1.

Za ovu vježbu koristi tablicu s 4 stupca. Napiši funkciju:

```
int najboljiKoridor(int teren[][4], int brojRedaka, int D);
```

redak	0	1	2	3
0	12	15	14	13
1	20	28	21	25
2	9	11	10	12
3	30	31	33	32

Za D = 4:

redak 0 ima razliku 15 - 12 = 3; redak 1 razliku 8; redak 2 razliku 3; redak 3 razliku 3. Sigurni su 0, 2 i 3. Budući da imaju jednaku najmanju razliku, funkcija vraća 0.

Tvoj algoritam prije pisanja koda mora odgovoriti na ova pitanja:

- Kako za jedan redak pronaći najmanju i najveću vrijednost?
- Kako iz toga odrediti je li redak siguran?
- Koje podatke treba pamtiti dok uspoređuješ više sigurnih redaka?
- Kako osigurati da kod izjednačenja ostane redak s manjim indeksom?

Obavezni dodatni testovi:

- D toliko malen da nijedan redak nije siguran -> rezultat mora biti -1;
- dva sigurna retka s različitim razlikama -> mora pobijediti manja razlika;
- dva najbolja retka s jednakom razlikom -> mora pobijediti manji indeks.

Ne pokušavaj problem riješiti nizom posebnih slučajeva. Traži podatke koje je dovoljno pamtiti dok jednom prolaziš kroz svaki redak.

DODATNI IZAZOV - KVAR U MREŽI SENZORA

U vektoru su kronološki zapisana mjerenja senzora. Senzor se smatra sumnjivim ako se pojave TRI UZASTOPNA mjerenja od kojih je svako strogo veće od prethodnog. Napiši funkciju koja vraća indeks trećeg mjerenja prvog takvog niza, a -1 ako takav niz ne postoji.

Samostalno odredi potpis funkcije.

Primjeri:

- {4, 6, 9, 3, 5} -> 2
- {8, 7, 9, 10, 2} -> 3
- {1, 2, 2, 3, 4} -> 4
- {5, 4, 3, 2} -> -1

Prije programiranja objasni zašto u primjeru {1, 2, 2, 3, 4} rezultat nije 2. Smisli vlastiti niz koji bi mogao otkriti pogrešku u rješenju koje koristi >= umjesto >.

Što predaješ?

Predaj trenutačno stanje rada na kraju vježbe. Za svaki obavezni problem predaj zasebnu .cpp datoteku.

- Na početku datoteke u komentaru: ime i prezime, naziv zadatka.
- Komentar: ULAZNI PARAMETRI -> ŠTO FUNKCIJA RADI -> ŠTO VRAĆA / MIJENJA.
- Barem jedan vlastiti test s očekivanim rezultatom određenim prije pokretanja.
- Za 3. zadatak u komentaru kratko opiši algoritam riječima prije programskog koda.
- Ako si koristio/la AI, napiši za što i što si samostalno provjerio/la.
- Ako zadatak nije dovršen, predaj postojeće stanje i napiši gdje si zapeo/la.

Samoprocjena prije predaje

☐ Mogu koristiti vector kao argument funkcije i znam kada mi ne treba poseban n.

☐ Mogu u 1D polju analizirati odnos susjednih elemenata.

☐ Mogu u 2D polju obraditi retke i usporediti rezultate obrade.

☐ Znam koristiti rubni slučaj za provjeru operatora > i >=.

☐ Mogu prekinuti obradu kada je traženi rezultat već pronađen.

☐ Mogu objasniti algoritam trećeg zadatka bez čitanja koda.

Moja procjena osobne sigurnosti (1-5): _____

Najviše trebam uvježbati: _____