

LV8 – Rekurzija

MODUL PROGRAMIRANJE | SIU1 – Funkcije i rekurzija | UTR | C++

Ishod učenja: Prepoznati i koristiti rekurziju u rješavanju problema.

Predviđeno trajanje: 2 školska sata (90 minuta) • Oblik rada: individualno ili u paru • Alati: C++ prevoditelj i uređivač koda

Cilj i upute

U ovoj vježbi napisat ćeš tri rekurzivne funkcije. Za svaku najprije odredi osnovni slučaj (kada se pozivanje zaustavlja), zatim rekurzivni korak (kako problem postaje manji). Rješenje treba koristiti rekurziju za ključnu obradu; petlje nisu dopuštene unutar traženih rekurzivnih funkcija. Smiješ koristiti string i vector iz STL-a. Za svaki zadatak izradi zaseban program ili jasno odvojene datoteke.

Prije pokretanja svakog programa rukom prati najmanje jedan primjer: zapiši slijed poziva i trenutak povratka iz rekurzije. U završnom osvrtu objasni što bi se dogodilo bez osnovnog slučaja.

Zadatak 1 – Digitalno zrcalo

UVODNI • približno 20 minuta

Digitalni sustav prikazuje broj znamenku po znamenku, ali od posljednje prema prvoj. Primjerice, za 1204 prikazuje 4021. Napiši rekurzivnu funkciju koja ispisuje znamenke zadanog nenegativnog cijelog broja obrnutim redoslijedom. Vodeće nule nastale obrtanjem ostaju vidljive (1200 → 0021).

Ulaz

Jedan cijeli broj n , $0 \leq n \leq 1\,000\,000\,000$.

Izlaz

Znamenke broja n u obrnutom redoslijedu, bez razmaka.

Primjeri

Ulaz	Izlaz
1204	4021
Ulaz	Izlaz
1200	0021
Ulaz	Izlaz
0	0

Uvjeti rješenja

- Obvezna rekurzivna funkcija `void zrcalo(unsigned int n)`.
- Funkcija ne smije koristiti petlju, string niti globalne varijable.
- Razmisli o osnovnom slučaju za jednoznamenasti broj i broj 0.

Objasni zašto rješenje koje najprije ispiše $n \% 10$, a zatim pozove funkciju za $n / 10$, radi drukčije od rješenja s obrnutim redoslijedom tih naredbi.

Zadatak 2 – Potraga kroz tunel

PRIMJENA • približno 25 minuta

Istraživački robot kreće se nizom prostorija. Svaka prostorija ima cijeli broj energetske jedinice (može biti i negativan). Robot želi znati koliko prostorija sadrži strogo pozitivnu energiju. Napiši rekurzivnu funkciju koja broji takve prostorije u vektoru.

Ulaz

U prvom retku n ($1 \leq n \leq 1000$). U drugom retku n cijelih brojeva od -1000 do 1000 .

Izlaz

Jedan cijeli broj: broj pozitivnih elemenata.

Primjeri

Ulaz	Izlaz
6 3 -2 0 7 -1 5	3
Ulaz	Izlaz
4 0 -5 -2 0	0

Uvjeti rješenja

- Obavezna funkcija `int brojPozitivnih(const vector<int>& a, int i)`, gdje je i indeks od kojeg se obrađuju elementi.
- Osnovni slučaj nastupa kada i dosegne `a.size()`.
- U funkciji ne koristiti petlju; svaki poziv obrađuje jedan element i preostali dio vektora.

Preoblikuj funkciju tako da umjesto broja pozitivnih elemenata vraća njihov zbroj. Koji se dio rekurzije mijenja?

Zadatak 3 – Put do cilja

NATJECATELJSKI TIP ZADATKA • približno 35 minuta

U računalnoj igri lik se nalazi na polju 0 i želi stići na polje n . U jednom potezu može prijeći za 1 ili za 2 polja. Koliko različitih nizova poteza vodi točno na polje n ? Redoslijed poteza je važan: za $n = 3$ mogući su $(1,1,1)$, $(1,2)$ i $(2,1)$. Ovakvi zadaci prebrojavanja načina čest su obrazac u natjecateljskom programiranju.

Ulaz

Jedan cijeli broj n , $0 \leq n \leq 30$.

Izlaz

Broj različitih nizova poteza.

Primjeri

Ulaz	Izlaz
3	3
Ulaz	Izlaz
4	5
Ulaz	Izlaz
0	1

Uvjeti rješenja

- Obvezna funkcija `long long nacini(int n)` s rekurzivnim pozivima.
- Odredi osnovne slučajeve: $n = 0$ znači jedan način (ne napraviti nijedan potez); za $n < 0$ nema načina.
- Rekurzivno poveži broj načina za $n - 1$ i $n - 2$.
- Zabilježi kako raste broj poziva pri povećanju n ; usporedi s iterativnim rješenjem.

Ako završiš ranije, dodaj memoizaciju pomoću `vector<long long>` kako se isti potproblem ne bi izračunavao više puta. Objasni razliku u učinkovitosti.

Završna refleksija (5–10 minuta)

1. Za svaki zadatak navedi osnovni slučaj i način smanjivanja problema.
2. U kojem zadatku se rezultat gradi pri povratku iz poziva, a u kojem se ispis događa prije sljedećeg poziva?
3. Zašto je naivna rekurzija u trećem zadatku sporija od prve dvije?
4. Koji bi zadatak bilo najlakše riješiti iterativno? Obrazloži.

Samoprocjena

Mogu...	Da	Još vježbam
prepoznati osnovni i rekurzivni slučaj	☐	☐
napisati rekurzivnu funkciju koja obrađuje broj ili vektor	☐	☐
pratiti slijed rekurzivnih poziva	☐	☐
objasniti zašto ponovljeni pozivi mogu usporiti algoritam	☐	☐

Predaja: tri .cpp datoteke (Z1.cpp, Z2.cpp, Z3.cpp) i kratki odgovori na refleksiju. Programi trebaju čitati samo zadani ulaz i ispisivati samo traženi rezultat, bez dodatnih poruka.

Napomena

Formativno pratiti: ispravan osnovni slučaj, napredovanje prema njemu, točnost i objašnjenje rekurzivnog koraka. Učenici koji trebaju potporu: dovoljno je dovršiti prva dva zadatka i opisati ideju trećeg.

Napredniji učenici: uvode memoizaciju.