

LV6 – FUNKCIJE: PODIJELI PROBLEM, POVEŽI RJEŠENJE

Programiranje • SIU1 • Ishod 1: Odrediti ulazne i izlazne parametre funkcionalnih cjelina

Trajanje	2 školska sata (UTR)
Način rada	Po mogućnosti individualno; ako dvoje učenika dijeli računalo, izmjena uloga nakon 1. zadatka.
Cilj	Samostalno projektirati i povezati više funkcija u jednom programu te razlikovati funkcije koje vraćaju vrijednost od funkcija tipa void.
Još ne koristimo	Polja i vektor kao parametre funkcija.
Predaja	Predati trenutačno stanje rješenja. Vježba služi učenju i kao dokaz rada; ne ocjenjuje se zasebno.

OPĆE UPUTE ZA RAD: RAZUMIJ → ODREDI ULAZE I JEDAN IZLAZ → OSMISLI FUNKCIJU → PREDVIDI → NAPIŠI → TESTIRAJ → DORADI

- Poželjno je raditi samostalno. Ako dijelite računalo, zamijenite se za tipkovnicom nakon prvog zadatka; svaki učenik osmišljava vlastiti test i mora moći objasniti cijelo rješenje.
- Za svaki obvezni zadatak prije pokretanja zapišite jedan vlastiti test s očekivanim rezultatom i razlogom odabira.
- Predajte vlastito trenutačno stanje rada do kraja sata, čak i ako program nije dovršen. Vježba se ne ocjenjuje zasebno; služi učenju i praćenju napretka prema ishodu.
- Al može objasniti poruku o pogrešci ili predložiti dodatni test, ali ne treba pisati gotov program umjesto vas. Zabilježite značajniju uporabu i prompt.

Prije kodiranja

Za svaki zadatak najprije u bilježnicu napiši:

- Koje manje funkcionalne cjeline prepoznaješ?
- Što svaka funkcija prima?
- Što svaka funkcija vraća – ili samo izvršava radnju?
- Koji će se rezultat jedne funkcije koristiti u drugom dijelu programa?
- Jedan vlastiti testni primjer i očekivani rezultat.

Tek nakon toga piši program.

ZADATAK 1 – SEMAFOR REZULTATA

Na školskom natjecanju učenik rješava tri zadatka. Za svaki zadatak dobiva cijeli broj bodova od 0 do 100. Ukupan rezultat je zbroj bodova. Učenik prolazi u finale ako ima najmanje 180 bodova ukupno i ni na jednom zadatku nema manje od 40 bodova.

Program mora biti podijeljen na najmanje tri funkcije:

- `int ukupno(int a, int b, int c)` – vraća ukupan broj bodova;
- `bool prolazi(int a, int b, int c)` – vraća informaciju prolazi li učenik u finale;
- `void ispisiRezultat(...)` – ispisuje ukupan rezultat i poruku `FINALE` ili `NE`.

Sam/a odluči koje parametre treba imati funkcija `ispisiRezultat`. Pokušaj izbjeći nepotrebno ponovno računanje onoga što je program već izračunao.

Ulaz: tri cijela broja. Izlaz: ukupan broj bodova i odluka.

Primjer ulaza	Očekivani ispis
75 82 51	208 FINALE
100 100 35	235 NE
60 60 60	180 FINALE

Vlastiti test mora provjeriti barem jednu graničnu vrijednost (40 ili 180).

Pitanje nakon rješenja: Bi li funkcija `ispisiRezultat` trebala sama ponovno računati zbroj? Obrazloži jednom rečenicom.

ZADATAK 2 – DIGITALNI SAT

Zadano je vrijeme u obliku sati, minuta i sekundi. Potrebno je ispisati vrijeme nakon točno jedne sekunde. Vrijeme je u 24-satnom obliku. Nakon 23:59:59 slijedi 00:00:00.

Rješenje organiziraj pomoću više funkcija. Obavezno koristi:

- barem jednu funkciju koja vraća vrijednost;
- funkciju tipa `void` za ispis vremena;
- funkciju bez parametara `void naslov()` koja ispisuje kratki naslov programa, npr. `SLJEDECA SEKUNDA`.

Sam/a odredi ostale funkcije i njihove parametre. Jedna moguća ideja je izdvojiti provjeru je li broj u dopuštenom rasponu, ali ne moraš slijediti taj prijedlog. Ne koristi polja ni vector.

Ulaz	Vrijeme nakon jedne sekunde
12 34 56	12:34:57
8 59 59	09:00:00
23 59 59	00:00:00

Za ispis vodeće nule možeš koristiti vlastitu logiku (npr. dodatni `cout << 0` kada je vrijednost manja od 10); nije potrebno koristiti posebne biblioteke za formatiranje.

Vlastiti test: odaberi primjer u kojem se mijenja sat, ali se ne prelazi u novi dan.

ZADATAK 3 – TAJNI TREZOR

Treći zadatak je namjerno zahtjevniji i rješenje zahtijeva rastavljanje problema na više jasnih funkcionalnih cjelina.

Trezor se otvara četveroznamenkastim kodom N ($1000 \leq N \leq 9999$). Kod je SIGURAN ako istodobno vrijede sva tri uvjeta:

- zbroj prve i četvrte znamenke jednak je zbroju druge i treće znamenke;
- sve četiri znamenke međusobno su različite;
- broj dobiven zamjenom prve i posljednje znamenke djeljiv je sa 7.

Za zadani N ispiši `SIGURAN` ako kod zadovoljava sva tri uvjeta, a inače `NIJE SIGURAN`. Uz poruku ispiši i koliko je od tri uvjeta kod zadovoljio.

Program mora sadržavati najmanje četiri funkcije, od kojih:

- najmanje dvije vraćaju vrijednost tipa `bool`;
- jedna vraća broj (npr. broj zadovoljenih uvjeta ili transformirani broj);
- jedna je tipa `void` i služi isključivo za završni ispis.

Ne propisujemo potpise funkcija. To je dio zadatka: sam/a odluči koje su funkcionalne cjeline smislene, koje podatke trebaju primiti i što trebaju vratiti. Ne koristi polja, vector ni string za izdvajanje znamenaka.

Kako pristupiti 3. zadatku

Prije programiranja riješi mali primjer na papiru:

$N = 1234 \rightarrow$ znamenke 1, 2, 3, 4

- Provjeri prvi uvjet.
- Provjeri jesu li sve znamenke različite.
- Konstruiraj broj nakon zamjene prve i posljednje znamenke.
- Provjeri djeljivost sa 7.
- Tek zatim odluči koje od tih radnji zaslužuju zasebnu funkciju.

Testni primjeri za 3. zadatak

Namjerno nisu navedeni očekivani izlazi za sve primjere. Prvi korak je da ih izračunaš ručno i zapišeš očekivanje prije pokretanja programa.

N	Očekivani rezultat	Svrha testa
1234	izračunaj prije pokretanja	osnovni primjer
2112	izračunaj prije pokretanja	ponovljene znamenke
7001	izračunaj prije pokretanja	nula unutar koda / transformacija
vlastiti primjer	obavezno predvidi	pokušaj pronaći SIGURAN kod ili primjer koji prolazi točno 2 uvjeta

DODATNI IZAZOV – ako završiš ranije

Bez mijenjanja uvjeta zadatka 3, preuredi program tako da `main()` sadrži što manje logike: neka uglavnom učitava podatak, pozove funkcije i pokrene završni ispis. Zatim objasni partneru ili nastavniku zašto je takav program lakše testirati.

AI kao tester, ne kao autor rješenja

Ako nastavnik dopusti uporabu AI-a, nakon što napišeš vlastito rješenje možeš zatražiti samo testni primjer koji bi mogao otkriti pogrešku. Nemoj tražiti gotovo rješenje.

Primjer upita: „Nemoj mi rješavati zadatak. Predloži jedan ulazni test koji bi mogao otkriti pogrešku u mojem programu i objasni samo što taj test provjerava. Nemoj izračunati očekivani rezultat.”

Što predaješ

- jednu `.cpp` datoteku za svaki riješeni zadatak;
- u komentaru na početku: ime/imena, nazive korištenih funkcija te za svaku kratko `ULAZ → IZLAZ`;
- barem jedan vlastiti test s očekivanim rezultatom za svaki zadatak;
- ako je korišten AI, jednu kratku bilješku za što je korišten.

Samoprovjera prije predaje

- ☐ Mogu objasniti zašto svaka moja funkcija postoji.
- ☐ Znam koje podatke svaka funkcija prima.
- ☐ Razlikujem funkciju koja vraća vrijednost od `void` funkcije.
- ☐ U programu imam više funkcija koje surađuju, a ne jednu veliku funkciju.
- ☐ Za zadatak 3 mogu objasniti algoritam i bez pokazivanja koda.

