TEST 1

1. Šta je Računar?

- **Računar** je elektronski uređaj koji memoriše i obrađuje podatke.

2. Iz čega se sastoji računar?

Računar se sastoji iz sledećih **komponenti**:

1. Centralna Procesorska Jedninica (CPU),
2. Glavna memorija,
3. Uređaj za skladištenje podataka,
4. Uređaji za unos podataka,
5. Uređaji za prikazivanje podataka,
6. Uređaji za komunikaciju.

Sve ove komponente su povezane sistemom koji se naziva Magistrala.

3. Komponente računara su povezane:

- magistralom podataka.

4. Šta je Računarski program?

- **Računarski program** (softver) je specifikacija tipova podataka I instrukcija za izvršavanje operacija koje računar koristi kako bi rešio postavljeni problem.

5. Šta je Programiranje?

- **Programiranje** je postupak kreiranja i razvijanja programa (softvera), ili drugim rečima, to je planiranje izvršavanja zadataka ili događaja. Vrši se pomoću programskih jezika.

6. Šta je Informacija?

- **Informacija** je saznanje preneto kroz prostor i vreme. Informacija se sastoji od podataka i značenja koja su dodeljena tim podacima.

7. Šta je Podatak?

- **Podaci** se mogu javiti u raznim oblicima: slova, reči, celi brojevi, … i sl.

8. Šta je programski jezik?

- **Programski jezik** je jezik za pisanje programa koje računar može izvršiti. Definisani su preko sintaksnih i semantičkih pravila koja opisuju njihovu strukturu i značenje.

9. Iz kojih koraka se sastoji Faza Rešavanja Problema?

Faza rešavanja problema se sastoji iz:

1. Analize i specifikacije – podrazumeva razumevanje algoritma,
2. Opšteg rešenja (algoritam) – podrazumeva specifikaciju neophodnih tipova podataka I logičke sekvence koraka koja rešava problem,
3. Verifikacije.

10. Iz kojih koraka se sastoji Faza implementacije?

Faza implementacije se sastoji iz:

1. Konkretnog rešenja – programa, tj. vrši se prevođenje algoritma u konkretan programski jezik,
2. Testiranja – gde se vrši provera da li računar pravilno prati zadate instrukcije.

TEST 2

1. Šta je mašinski jezik?

   - **Mašinski jezik** čini skup ugrađenih jednostavnih mašinskih instrukcija u binarnom obliku, razumljiv je računaru ali ne i ljudima.

2. Šta je mašinska instrukcija?

   - **Mašinska instrukcija** se sastoji od operacionog koda (opcode) i operanada koji određuju nad kojim podacima se obavlja ta operacija (registri, mem-mokacije, ...).

3. Šta je Asemblerski jezik i čemu služi?

   - **Asemblerski jezik** se smatra jezikom niskog nivoa, jer je po prirodi blizak mašinskom jeziku I zavistan je od mašine, tj. računara.

4. **Programski (ili poznati) jezici treće generacije** …

   - ne zavise od racunara.

5. Šta od navedenog je tačno?

   - **Asemblerski jezik je jezik niskog nivoa**?
   - Asembelrski jezik je **blizak mašinskom jeziku** i koristi prevodilac Asembler za prevođenje vlastitih instrukcija na mašinski jezik.

6. Koje **4 Paradigme u razvoju softvera** postoje?

   Paradigme su: 1. Imperativna (proceduralna) paradigma,
   2. Deklarativna paradigma,
   3. Funkcionalna paradigma,
   4. Objektno-orjentisana paradigma.

7. Opisati **Imperativnu paradigmu**.

   - Imperativna paradigma rešava problem primenom odgovarajućeg algoritma I potrebnih podataka. Ona predstavlja tradicionalan način programiranja.

8. Opisati **Deklarativnu paradigmu**.

   - Deklarativna paradigma zahteva od programera da opiše problem koji ima, a ne algoritam za njegovo rešavanje.

9. Opisati **Funkcionalnu paradigmu**.

   - Funikcionalna paradigma dovodi do rezultata na osnovu unosa određenih podatak. Rezultat se dovodi sekvencijalnom primenom niza jednostavnih funkcija.

10. Sistem kao skup objekata koji interreaguju i izvode akcije za rešavanje konkretnog problema predstavljen je:

    - **Objektno-orjentisanom paradigmom** razvoja softvera.

11. Koja **tri osnovna zahteva** mora da zadovolji programski jezik?

    Zahtevi su:
    1. Programski jezik mora da bude univerzalan,
    2. Mora da postoji mogućnost implementacije programskog jezika na računaru,
    3. Programski jezik treba da ima prihvatljivo efikasnu implementaciju.

12. **Jednostavna instrukcija** (double pi = 3.14) napisana u Javi ili C++ jeziku, prevodi se na skup mašinskih instrukcija koje ...
    - rezervišu potrebnu memoriju,
    - smeštaju broj na odgovarajuću memorijsku lokaciju,
    - beleže tu lokaciju za kasniju upotrebu.

13. Koja je razlika između **Sintakse** i **Semantike** programskog jezika?
    - **Sintaksa** programskog jezika se odnosi na formu programa a **Semantika** se odnosi na značenje programa.

14. Koji su osnovni tipovi instrukcija u nekom programskom jeziku?
    Osnovni tipovi instrukcija su:
    1. sekvenca,
    2. selekcija,
    3. petlja,
    4. potprogram.

15. Šta su Petlje?
    - **Petlja (loop)** je ponavljajuća kontrolna struktura koja ponavlja skup iskaza, sve dok se ne zadovolji neki uslov

16. Šta je Potprogram?
    - **Potprogram (subprogram)** je imenovana sekvenca instrukcija napisana odvojeno od glavnog programa.

17. Koja su tri tipa iskaza koja se javljaju u programu?
    Tipovi iskaza su:
    1. Deklarativni iskazi,
    2. Imperativni iskazi,
    3. Komentari.

18. Šta su promenljive a šta su konstante?
    - **Promenljiva** je naziv memorijske lokacije sa podatkom promenljive vrednosti.
    - **Konstanta** je podatak koji ne menja vrednost u toku izvršenja programa.

19. Ukratko opisati imperativnu paradigmu …
    - **Imperativna paradigma** rešava problem primenom odgovarajućeg algoritma i potrebnih podataka.

20. Koja je razlika između asemblera i kompajlera?
    - **Kompajler** je program prevodilac (translator) koji prevodi program koji sadrži primitive višeg nivoa u program predstavljen mašinskim jezikom.
    - **Asembler** prevodi asemblerski jezik u mašinske instrukcije.

TEST 3

1. Šta je Algoritam?

- **Algoritam** je opis rešavanje nekog problema, drugim rečima to je, pismeni ili usmeni opis sekvence akcija koje se primenjuju nad objektima. To je konačna I precizno definisana procedura (niz pravila) kojom se ulazne vrednosti transformišu u izlazne.

2. Na koji način se predstavlja algoritam?

Algoritmi se predstavljaju:

1. prirodnim jezikom,
2. grafički dijagramom toka,
3. tekstualno pomoću pseudokoda,
4. odgovarajućim programskim jezikom

3. Koje su osobine algoritma?

Osobine algoritma su:

1. diskretnost,
2. rezultativnost,
3. determinisanost,
4. masovnost.

4. Šta je rešavanje problema?

- **Rešavanje problema** se vrši pomoću računara uzimanjem informacija od korisnika, obradom ulaza i proizvodnjom neke vrste izlaza u vidu slike, teksta, zvuka.

5. Kojih 6 osnovnih korak treba izvršiti da bi stigli do rešenja?

Koraci su:

1. Razumevanje problema,
2. Formulisanje modela,
3. Razvoj algoritma,
4. Pisanje programa,
5. Testiranje rešenja,
6. Evaluacija rešenja.

6. Da bismo bili sigurni da smo potpuno razumeli problem neophodno je da sami sebi postavimo neka pitanja. Navesti neka od tih pitanja.

Pitanja su:

- Koji ulazni podaci-informacije su nam dostupni?
- Šta ti podaci predstavljaju?
- U kom formatu su podaci zapisani?
- Da li slučajno nešto nestaje?
- Da li imamo sve što je potrebno da bismo rešili problem?
- Koju izlaznu informaciju pokušavamo da kreiramo?
- U kom obliku želimo da predstavimo podatke?
- Šta sve moramo da izračunamo da bismo došli do rešenja?

7. Šta je **Pseudokod**?

- Pseudokod je prostija i konciznija sekvenca instrukcija (najčešće pisanih na engl. jeziku) koju koristimo u cilju rešavanja problema.

8. Šta su **prednosti pseudokoda**?

Prednosti su:

1. Lakše je pseudokod napisati na listu papira, nego dijagram toka koji može da bude prevelik,
2. Pseudokod može biti napisan na način koji je sličan realnom programskom kodu,
3. Manje vremena je potrebno za pisanje pseudokoda nego za dijagram toka.

9. Šta su **BAG**-ovi?

- Bag-ovi, tj. kvarovi su problemi, odnosno greške sa progrmom koje izazivaju da program prestane sa radom ili da daje netačna rešenja.

10. Šta je **Debagovanje**?

- Proces pronalaženja i otkalnjanja grešaka u programu se naziva debagovanje (debuging).

11. Šta je apstrakcija?

- **Apstrakcija** je ideja korišćenja najjasnijeg algoritma koji nema u sebi puno nepotrebnih detalja.

12. Koja je uloga apstrakcije u rešavanju problema?

- **Uloga apstrakcije** je da se proizvede što je moguće prostiji algoritam, koji istovremeno I lak za razumevanje.

13. Koja je uloga podalgoritma?

- Uloga **podalgoritma** je da omogući razlaganje ukupnog problema (algoritma) koji se rešava na manje probleme koje je jednostavnije rešiti.

14. Šta su Funkcije a šta su Procedure?

- **Funkcija** je podalgoritam koji kao rezultat vraća neki tip objekta ili vrednost.
- **Procedura** je podalgoritam koji ne vraća pozivaocu nikakvu vrednost.

15. Koja je **uloga postavljanja pitanja** u procesu programiranja?

- Uloga je da omogući programeru da bude svestan strategije koju treba da korist kako bi rešio problem.

TEST 4

1. Kada kažemo da je broj prost?

   - **Prosti brojevi** su svi prirodni brojevi deljivi bez ostatka samo sa brojem 1 i sa samim sobom a strogo veći od 1.

2. Šta je dinamičko programiranje?

   - **Dinamičko programiranje** je pristup koji itbegava ponovno izračunavanje rešenja koja su već izračunata.

3. Šta je dijagram toka?

   - **Dijagram toka** (flowchart) se koristi sa ciljem da se vizualno prikaže tok ili kontrola nekog algoritma, kako se on izvršava korak po korak.

4. Koji su osnovni **elementi dijagrama toka**?

   Elementi su: 1. operacija dodele,
   2. učitavanje podataka,
   3. štampanje podataka,
   4. uslovni operator,
   5. operator ponavljanja (petlja).

5. Opisati **operatore** dijagrama toka **za učitavanje i štampanje** podataka.

   - Ulaz (učitavanje podataka) – Operacija učitavanja postavlja promenljivu na vrednost koju je uneo korisnik. Predstavlja se pomoću paralelograma i strelice koja pokazuje ka njemu.
   - Izlaz (štampanje podataka) – Izlazni operator prikazuje vrednost odgovarajuće promenljive ili štampa poruku. Predstavljen je pomoću paralelograma i strelice koja pokazuje udesno.

6. Opisati **uslovni operator** dijagrama toka (isti je kao i **operator ponavljanja**-petlje).

   - Uslovni operator određuje smer u kome će teći algoritam. Predstavljen je pomoću dijamanta.

7. Šta je pohlepni algoritam?

   - **Pohlepni algoritam** u svakom koraku bira naizgled najbolje rešenje u tom trenutku, bez obzira da li će to uticati na tačnost konačnog rešenja problema.

8. Šta je iscrpna pretraga? Koji se još nazivi koriste za iscrpnu pretragu u literaturi?

   - **Iscrpna pretraga** je opšta tehnika rešavanja problema koja se sastoji od sistematičnog nabrajanja svih kandidata kao mogućih rešenja i proverava da li svaki zadovoljava rešenje. Nazivi su još i **Brute-Force** pretraga ili **gruba sila**.

9. Šta je podeli i osvoji pristup?

   - **Podeli i osvoji** (zavadi pa vladaj) predstavlja algoritam u kojem se veći problem rasčlanjuje na više manjih koji su jednostavniji za sagledavanje I pojedinačno rešavanje.

10. Šta je **linearno pretraživanje?**

    - Metod linearnog pretraživanja je metoda koji upoređuje ključ sa elementima niza. Pretraga se izvršava sve dok se ključ pretrage ne poklopi sa nekim elementom niza.

11. Šta je **Palindrom**?

    - Reč je Palindrom ako se čita isto sa obe strane.