ФОН

Пријемни испит за студијски програм Софтверско инжењерство и рачунарске науке 2021.

Шифра задатака:	6	1	1	4	1	1

---

1. У Јави се нит може направити на два начина:
   a. реализацијом интерфејса *Thread* или проширењем класе *Runnable*
   b. реализацијом интерфејса *Runnable* или проширењем класе *Thread*
   c. реализацијом интерфејса *Runnable* или проширењем класе *Execute*
   d. реализацијом интерфејса *Execute* или проширењем класе *Thread*
   e. реализацијом интерфејса *Thread* или проширењем класе *Execute*
   n. не знам

2. Методе *wait*(), *notify*() и *notifyAll*() могу се једино позвати из:
   a. *static* метода
   b. синхронизованих метода
   c. *public* метода
   d. перзистентних метода
   e. простих метода
   n. не знам

3. Сокет је механизам који омогућава комуникацију између:
   a. програма
   b. објеката и класа истог програма
   c. табела базе података
   d. класа
   e. екранских форми
   n. не знам

4. Када се нит креира са *new* оператором:

```
nit = new Thread(this,"Nova nit")
```

   нит прелази у стање:

   a. ново
   b. свршено
   c. извршно
   d. блокирано
   e. почетно
   n. не знам

5. Јава програм успоставља конекцију са базом података преко следеће наредбе:
   a. `import connection.java.sql.*;`
   b. `connection.jdbc:mysql://127.0.0.1:3306/predmet;`
   c. `Connection con = DriverManager.getConnection(dbUrl,user,pass);`
   d. `String connection = "SELECT * FROM Predmet";`
   e. `Class.forName("com.mysql.jdbc.Driver")`
   n. не знам

6. Кориснички интерфејс представља:
   a. динамичку репрезентацију софтверског система
   b. архитектурну репрезентацију софтверског система
   c. улазно-излазну репрезентацију софтверског система
   d. формалну репрезентацију софтверског система
   e. објектну репрезентацију софтверског система
   n. не знам

7. Из којих секција се састоје уговори код системских операција:

a. Операција, Веза са СК, Вредносна ограничења, Структурна ограничења
b. Операција, Веза са СК, Листа аргумената, Тело операције
c. Операција, Актори, Листа аргумената, Тело операције
d. Операција, Актори, Предуслови, Постуслови
e. Операција, Веза са СК, Предуслови, Постуслови
n. не знам

8. За податке се каже да су конзистентни ако задовољавају:

a. вредносна и системска ограничења дефинисана над тим подацима
b. вредносна и структурна ограничења дефинисана над тим подацима
c. пројектна и структурна ограничења дефинисана над тим подацима
d. пројектна и системска ограничења дефинисана над тим подацима
e. типска и програмска ограничења дефинисана над тим подацима
n. не знам

9. Из чега се састоји софтверски систем?

a. Софтверски систем се састоји од објеката и типова
b. Софтверски систем се састоји од нити и сокета
c. Софтверски систем се састоји од апликационе логике и датотека
d. Софтверски систем се састоји од атрибута и системских операција
e. Софтверски систем се састоји од класа и табела
n. не знам

10. Шта је подржавост софтверског система?

a. Подржавост система се односи на лакоћу његовог прилагођавања и одржавања, интернационализацију и начина конфигурисања
b. Подржавост система се односи на лакоћу одржавања интерфејса система и брокера базе података
c. Подржавост система се односи на лакоћу одржавања апстрактних метода које омогућавају извршење системских операција система
d. Подржавост система се односи на лакоћу одржавања вредносних и структурних правила интегритета система
e. Подржавост система се односи на лакоћу његовог прилагођавања, поузданости и употребљивости
n. не знам

11. Друго Боемово правило (Boehm's second low) гласи:

a. уколико се не уоче грешке у току дефинисања захтева, исте се веома тешко могу уклонити у каснијим фазама развоја програма
b. прављење прототипова значајно смањује могуће грешке код дефинисања захтева и његовог развоја, нарочито код дефинисања корисничког интерфејса
c. сценарија СК не треба да буду у међусобној интеракцији
d. недостаци код дефинисања захтева су основни разлог могућег неуспеха у развоју пројекта (програма)
e. захтев за извршење једне или више системских операција се не одиграва континуално него дискретно
n. не знам

12. Нефункционални захтеви су:

a. употребљивост, трансијентност, функционалност
b. транспарентност, поузданост, подржив ост
c. употребљивост, конзистентност, подржив ост
d. транспарентност, конзистентност, перзистентност
e. употребљивост, поузданост, подржив ост
n. не знам

13. Погледати кôд који је дат испод. Шта је потребно додати на означено место (уместо ####) да би била правилно декларисана класа са називом "mojprelepiauto" и то тако да буде у складу са неписаним правилом (конвенцијом) о давању назива класама:

```
class #### {
}
```

a. `mojprelepiauto`
b. `moj_prelepi_auto`
c. `MojPrelepiAuto`
d. `Moj_Prelepi_Auto`
e. `mojPrelepiAuto`
n. не знам

14. Погледати кôд који је дат испод (број линије кôда је дат уз леву ивицу). Шта је потребно додати и у коју линију кôда, да би кôд био без синтаксних грешака?

```
1
2 public class Osoba {
3      /*Ova klasa predstavlja osobu
4
5 }
```

a. додати `//` у прву линију кôда
b. додати `/*` у четврту линију кôда
c. додати `*/` у четврту линију кôда
d. додати `//` у пету линију кôда
e. не треба ништа додавати, кôд нема синтаксних грешака
n. не знам

15. Погледати кôд који је дат испод (број линије кôда је дат уз леву ивицу). Шта је потребно додати и у коју линију кôда, да би кôд био без синтаксних грешака?

```
1
2 public class Osoba {
3      //Ova klasa predstavlja osobu
4
5 }
```

a. додати `//` на прву линију кôда
b. додати `/*` на четврту линију кôда
c. додати `*/` на четврту линију кôда
d. додати `//` на пету линију кôда
e. не треба ништа додавати, кôд нема синтаксних грешака
n. не знам

16. Погледати кôд који је дат испод. Шта је потребно додати на означено место (уместо ####) да би био правилно декларисан атрибут са називом "stalnaadresa" и то тако да буде у складу са неписаним правилом (конвенцијом) о давању назива атрибутима:

```
class Osoba {
String ####;
}
```

a. `Stalnaadresa`
b. `stalna_adresa`
c. `StalnaAdresa`
d. `stalnaAdresa`
e. `STALNA_ADRESA`
n. не знам

17. Погледати кôд две класе (`Klasa1` и `Klasa2`) који је дат испод. Да ли је кôд синтаксно исправан, и ако није, шта је потребно исправити?

```
package paket1;
public class Klasa1 {
	protected int atribut1;
}

package paket1.paket2;
import paket1.Klasa1;
public class Klasa2 {
	public static void main(String[] args) {
		Klasa1 k1 = new Klasa1();
		k1.atribut1 = 55;
	}
}
```

a. треба додати јавни конструктор у класу `Klasa1`
b. треба додати `"import paket1.paket2.Klasa2;"` у класу `Klasa1`
c. атрибут `atribut1` треба декларисати као јаван (`public`)
d. атрибут `atribut1` треба декларисати као приватни (`private`)
e. кôд је синтаксно исправан, ништа не треба исправљати
n. не знам

18. Погледати кôд који је дат испод. Шта ће се десити кад се покрене `main()` метода?

```
class Osoba {
	String ime;
}

class Test {
	public static void main(String[] args) {
		Osoba o1 = null;
		Osoba o2 = null;
		o1.ime = "Pera";
		o2.ime = "Mika";
		o2 = o1;
		System.out.println(o2.ime);
	}
}
```

a. на екрану ће се исписати `"Pera"`
b. на екрану ће се исписати `"Mika"`
c. на екрану ће се исписати `"Pera Mika"`
d. на екрану ће се исписати `"Mika Pera"`
e. Јава пријављује грешку
n. не знам

19. Погледати кôд који је дат испод. Који ће бити садржај листе чланови из класе `Grupa` када се изврши додавање три члана (Мика, Пера, Зика) из `main()` методе?

```
import java.util.LinkedList;
public class Grupa {
   LinkedList<String> clanovi = new LinkedList<String>();

   public void dodajClana(String clan) {
         if (clanovi.size() > 0)
               clanovi.add(clanovi.size(), clan);
   }

   public static void main(String[] args) {
         Grupa g = new Grupa();
         g.dodajClana("Mika");
         g.dodajClana("Pera");
         g.dodajClana("Zika");
   }
}
```

a. `["Mika", "Pera", "Zika"]`
b. `["Zika", "Pera", "Mika"]`
c. `["Mika", "Mika", "Mika", "Pera", "Pera", "Pera", "Zika", "Zika", "Zika"]`
d. `["Zika", "Zika", "Zika", "Pera", "Pera", "Pera", "Mika", "Mika","Mika"]`
e. `[]` – листа ће бити празна
n. не знам

20. Погледати кôд који је дат испод. Шта ће метода `ispisiFNiz()` исписати на екрану када се изврши?

```
public class FNiz {
   public static void ispisiFNiz() {
         int prvi = 0, drugi = 1;
         System.out.print(prvi + " ");
         for (int i = 1; i <= 9; i++) {
               System.out.print(drugi + " ");
               int pom = prvi + drugi;
               drugi = pom;
               prvi = drugi;
         }
   }
}
```

a. `0 0 0 0 0 0 0 0 0 0`
b. `1 1 1 1 1 1 1 1 1 1`
c. `0 1 1 2 4 8 16 32 64 128`
d. `0 1 1 2 3 5 8 13 21 34`
e. неће исписати ништа на екрану
n. не знам

21. Погледати кôд који је дат испод. Шта ће метода `ispisiFNiz()` исписати на екрану када се изврши?

```
public class FNiz2 {
   static void ispisiFNiz() {
          int prvi = 0, drugi = 1;
          System.out.print(prvi + " ");
          do {
                System.out.print(drugi + " ");
                int pom = prvi;
                prvi = drugi;
                drugi = prvi + drugi;
          } while (drugi >= 100);
   }
}
```

a. `0`
b. `0 1`
c. `0 1 1 2 3 5 8 13 21 34 55 89`
d. `0 1 1 2 4 8 16 32 64 128`
e. неће исписати ништа на екрану
n. не знам

22. Погледати кôд који је дат испод. Да ли постоји нека вредност за промењиву **a** (неки цео број) за коју обе методе (`proveriBroj()` и `proveriBroj2()`) увек враћају `true`?

```
class ProveraBroja {
   boolean proveriBroj(int a) {
          if (a<100 || a > 200) return true;
          return false;
   }

   boolean proveriBroj2(int a) {
          if (a>=99 && a < 50) return true;
          else   return false;
   }
}
```

a. `99`
b. `100`
c. `50`
d. `201`
e. не постоји таква вредност
n. не знам

23. Погледати кôд који је дат испод. Шта се дешава када се позове метода `ispis()`?

```
public class Niz {
      int[] niz = new int[10];
      void ispis() {
      System.out.println(niz[10]);
      }
}
```

a. исписује се 0 на екрану
b. исписује се `null` на екрану
c. Јава баца изузетак `NullPointerException`
d. Јава баца изузетак `ArrayIndexOutOfBoundsException`
e. Јава баца изузетак `ArithmeticException`
n. не знам

24. Погледати кôд који је дат испод. Шта се дешава када се изврши `main()` метода?

```
public class Niz {
   int[] niz;

   public Niz(int n) {
         niz = new int[n];
   }

   void ispisi() {
         System.out.println(niz[0]);
   }

   public static void main(String[] args) {
         Niz n = new Niz(-5);
         n.ispisi();
   }
}
```

a. исписује се 0 на екрану
b. исписује се `null` на екрану
c. Јава баца изузетак `NullPointerException`
d. Јава баца изузетак `ArrayIndexOutOfBoundsException`
e. Јава баца изузетак `NegativeArraySizeException`
n. не знам

25. Следећим *SQL* наредбама креиране су табеле *STUDENT* и *SMER*, а потом су попуњене подацима:

```
CREATE TABLE SMER(
  S#      tinyint not null primary key,
  Naziv   nvarchar(24) not null);

CREATE TABLE STUDENT(
  I#      smallint not null primary key,   -- број индекса
  Ime     nvarchar(7) not null,
  Prezime nvarchar(7) not null check (Prezime LIKE '[A-Ш]%'),
  Smer#   tinyint null foreign key references SMER(S#));

INSERT INTO SMER VALUES
  (10, 'Софтверско инжењерство'),(20, 'Информациони системи'),
  (80, 'Управљање производњом'), (60, 'Управљање пословањем');

INSERT INTO STUDENT VALUES
  (17002, 'Ана',   'Костић', NULL), (17014, 'Ана',   'Марић',  NULL),
  (17008, 'Анка',  'Анић',   NULL), (16002, 'Аница', 'Барић',    10),
  (16014, 'Мара',  'Илић',     20), (16008, 'Мила',  'Јовић',    60),
  (15002, 'Аца',   'Костић',   10), (15014, 'Мома',  'Којић',    20),
  (15008, 'Јова',  'Кун',      60), (14002, 'Лаза',  'Марић',    60),
  (14014, 'Јова',  'Киш',      10);
```

Дати су следећи упити:

У-1:

```
SELECT S#, Ime, Prezime, Smer#
  FROM STUDENT ST
  WHERE EXISTS (
    SELECT * FROM SMER SM
    WHERE ST.Smer# = SM.S#
    AND Naziv LIKE 'Софтверско инжењерство');
```

У-2:

```
SELECT S#, Ime, Prezime, Smer#
FROM STUDENT
WHERE Smer# = (
SELECT S#
FROM SMER
WHERE Naziv LIKE 'Софтверско инжењерство');
```

У-3:

```
SELECT I#, Ime, Prezime, Smer#
FROM STUDENTS JOIN SMERST ON S.S#=ST.Smer#
WHERE Naziv LIKE 'Софтверско инжењерство';
```

У-4:

```
SELECT I#, Ime, Prezime, Smer#
FROM (
SELECT S#, Naziv FROM SMER
WHERE Naziv LIKE 'Софтверско инжењерство') SM
INNERJOIN STUDENT ST ON Smer# = SM.S#;
```

Наведите оне упите који враћају исте резултате:

a. У-1, У-2
b. У-2, У-4
c. међу наведеним упитима не постоје оне који враћају исте резултате
d. У-1, У-2, У-4
e. У-2, У-3
n. не знам

26. Шта је схема (*database schema*) у *T-SQL*-у:

a. именована група логичких повезаних објеката у бази података, која обједињује све објекте који деле исти именски простор (*namespace*)
b. скупа апликационих подсхема које садржи подмоделе података прилогођене корисницима базе
c. све што је претходно наведено
d. начин физичког груписања табела на једном екстерном меморијском носиоцу
e. ништа од претходно наведеног
n. не знам

27. Сваки модел података састоји се од:

a. колекције концепата за дефинисање структуре система (стања система)
b. колекције операција за манипулацију концептима структуре (промена стања, приказ стања)
c. колекције правила интегритета који су засновани на вредносним ограничењима
d. свега што је претходно наведено
e. онога што је наведено под а) и б)
n. не знам

28. Кључ релације је подскуп њених атрибута који има следеће особине:

a. не постоје било које две н-торке са истом вредношћу атрибута који чине кључ (особина јединствености)
b. ако се изостави било који атрибут из кључа, губи се особина јединствености (особина нередундантности)
c. наведене под а) и б)
d. не постоје било које две н-торке са истом вредношћу атрибута који чине кључ, јер сви атрибути који чине кључ морају бити *UNIQUE*(особина уникатности)
e. ништа од претходно наведеног
n. не знам

29. Приликом извршавања *SQL* упита (*query*) систем за управљање базама података извршава следеће кораке:

a. парсирање упита, валидација упита, генерисање план извршења упита, оптимизација плана извршења, извршење оптимизованог плана
b. валидација упита, оптимизација упита, генерисање плана извршења, извршење плана
c. валидација упита, парсирање упита, оптимизација упита, генерисање оптимизованог плана извршења пита, извршење упита
d. парсирање упит, оптимизација упита, генерисање план извршења упита, валидација плана извршења, извршење оптимизованог и валидираног плана
e. ништа од претходно наведеног
n. не знам

30. Ако се користе табеле *STUDENT* и *SMER*, из 25. задатка, извршењем упита:

```
SELECT S.I#, S.Ime, S.Prezime, Naziv Naziv
  FROM STUDENT S
  RIGHT OUTER JOIN SMER M ON Smer# = M.S#
ORDER BY S# DESC;
```

у првом реду резултата биће приказано:

a. `16002 Аница Барић Софтверско инжењерство`
b. `14002 Лаза Марић Управљање пословањем`
c. `14014 Јова Киш Софтверско инжењерство`
d. ништа, зато што је упит синтаксно неисправан
e. `NULL NULL NULL Управљање производњом`
n. не знам