Priprema za studentsku praksu

Zadatci – Software DEVELOPMENT Engineer

v2.1

# Teorijski dio

Prođite kroz listu pojmova iz prvog stupca tablice ispod s ciljem da upoznate, proučite i razumijete što je moguće više pojmova.

|  |  |
| --- | --- |
| **PROGRAMSKI JEZIK JAVA** | |
| **Uvod** | |
| Klasa | Predložak za objekte. Propisuje atribute (čuvanje stanja) i sadrži definicije metoda koje se nad objektom mogu pozvati. |
| Objekt | Instanca svoje klase koja ima identitet, stanje i ponašanje. Stanje je trenutna vrijednost dodijeljena atributima, ponašanje se određuje metodama koje se pozivaju nad objektom a identitet je određen jedinstvenim memorijskim zapisom. |
| Metoda | Kod koji je napisan u nekoj klasi, ima ime i može biti pozvana. Određuje ponašanje objekta. |
| Varijabla | Spremnik u radnoj memoriji kojem je dodijeljeno ime i vrsta podatka koji se u nju sprema. (U Javi imamo primitivne i referentne vrste) Lokalna varijabla se ne inicijalizira automatski za razliku od atributa. |
| Sučelje | Omogućuje interakciju između čovjeka i stroja/programa |
| Nasljeđivanje | Služi da bi stvorili nove klase ekstendanjem postojeće klase te da možemo koristiti metode i polja roditeljske klase (ili dodati nove metode i polja). IS-A odnos |
| Overload | Dozvoljava da klasa ima više metoda istog imena ali različitih parametre. |
| API | Application Programming Interface – služi za komunikaciju između više aplikacija |
| JDK | Java Development Kit – sadrži JRE + alate za developere . Implementacija neke od Java Platformi (Standard,Enterprise ili Micro) I ostalih resursa npr. archiver,compiler …. |
| JRE | Java Runtime Environment – set softverskih alata za razvijanje Java aplikacija. Implementacija JVM-a , sadrži libraries I ostale datoteke koje JVM koristi u runtime-u. |
| JVM | (Java Virtual Machine) – virtualna jer ne postoji fizički , zapravo je interpreter. JVM je ta koja zove main metodu i zadužena je za izvršavanje java programa red po red. Zbog JVM su java aplikacije WORA (Write Once Run Anywhere).  Učitava kod, provjerava kod, izvršava kod i pruža okruženje za vrijeme rada. |
| jar | Java Archive – format datoteka baziran na ZIP-u |
| j2se | Java platforma, standardno izdanje |
| j2ee | Java platforma, enterprise izdanje |
| OpenJDK vs OracleJDK | Razlika u licenciranju, OpenJDK je open source Java sa GNU General Public License licencom. OracleJDK zahtjeva komercijalnu licencu |
| Maven | Build i dependency management tool za java projekte -koristimo za standardiziranje dependency management-a |
| Ant | (“Another Neat Tool”) - java library koja se koristi za automatizaciju build procesa Java aplikacija |
| Gradle | Open source dependency management and build automation tool – dobre stvari Ant-a I Mavena |
| Koja su prva 4 byte-a svake klase (class file-a) u hex formatu i zašto? | 0xCAFEBABE |
| **Konvencija naziva** | |
| Naziv klasa | Imenice – prvo slovo veliko, trebaju biti jednostavna i deskriptivna |
| Naziv metoda | Glagoli – prvo slovo malo , prvo slovo iduće riječi veliko -> getColor(); |
| Naziv varijabli | Trebaju biti kratki ali značajni(razumljivi), - isto kao kod metoda mala pa velika slova npr. int mySize; |
| **Tipovi podataka** | |
| Primitivni tipovi | 8 primitivnih tipova podataka (byte, short, int, long, float, double, boolean, char) , definirano u samom jeziku i rezervirani keywordi |
| Referencirajući tipovi | Subklasa  java.lang.Object, za razliku od primitivnih koji sremaju stvarnu vrijednost referencirajući tipovi spremaju adresu objekta u memoriji npr. String, Scanner,Interface |
| **Operatori** | |
| Aritmetički | + - \* / % ++ -- |
| Relacijski | == equal to  != not equal to  > greater than  >= greater than or equal to  < less than  <= less than or equal to |
| Operatori uvjeta | && Conditional-AND  || Conditional-OR  ? if-then-else  ! not |
| Operatori pridruživanja | = += -= \*= /= %= |
| **Kontrola toka: odlučivanje** | |
| if | Izvrši ako je zadovoljen uvjet |
| if – else | Izvrši ako je zadovoljen uvjet , ako nije izvrši nešto drugo |
| if – else if – else | Provjeri uvijet, kao je zadovoljen izvrši, zatim provjeri n else if blokova(uvijeta) svaki koji zadovolji izvrši i nastavi na idući, na kraju izvrši ono što je u else |
| switch – case | Switch expression je evaluiran jednom i onda se ta vrijednost uspoređuje sa svakim case-om i izvršava se samo odgovarajući blok  break i default su opcionalni |
| **Kontrola toka: petlje** | |
| for | Služi za iteraciju nekog raspona vrijednosti  for (*initialization*; *termination*; *increment*) {  *statement(s)*  }  For-Each  for (String person : persons) {  }  Ne koristi se kad nam treba iterator (pošto for-each sakriva iterator) |
| while | Koristimo kad ne znamo broj iteracija i nije nam bitno da se izvrši barem jednom |
| do – while | Koristimo kad ne znamo broj iteracija i bitno nam je da se izvrši barem jednom |
| **Kontrola toka: bezuvjetni skok** | |
| break | Izlaz iz petlje |
| continue | Za prekid jedne iteracije petlje |
| return | Za vraćanje vrijednosti (jedan return value po metodi) ili izlaz iz metode |
| **Kontrola toka: nepravilnosti** | |
| try – catch | U try blok stavljamo kod koji bi mogao baciti „exception” a u catch blok ide kod koji se izvrši ako se try blok nije zbog nekog razloga mogao očekivano izvršiti |
| try – catch – finally | Finally služi da se kod koji je u finally bloku izvrši čak i ako je došlo do exceptiona |
| try – finally | Uvijek se izvršava finally blok nakon try  **try** {  *// Block of code with multiple exit points*  }  **finally** {  *// Block of code that is always executed when the try block is exited,*  *// no matter how the try block is exited*  } |
| throw | Koristimo za „bacanje“ exceptiona ili errora |
| **Ključne riječi** | |
| abstract | Koristi se za postizanje apstrakcije (klase i metode) , abstract class se ne može koristiti za kreiranje objekata, abstract metode nemaju specificiranu implementaciju u super klasi, sub klase ih overrideaju , koriste se u abstract klasi |
| assert | Koristi se za provjeravanje uvjeta , ako uvjet nije zadovoljen izbacit ce AssertionError |
| boolean | Primitivni tip / true ili false |
| break | Izlaz iz petlje |
| byte | Primitivni tip koji može „držati“ vrijednosti od -128 do 127 |
| case | Dio switcha (vrijednost expressiona se uspoređuje sa vrijednosti case-a) |
| catch | Ukoliko dode do exceptiona izvršava se kod u catch-u |
| char | The char data type is a single 16-bit Unicode character. It has a minimum value of '\u0000' (or 0) and a maximum value of '\uffff' (or 65,535 inclusive). |
| class | Keyword koji se koristi za deklaraciju nove Java klase |
| const | Ne koristi se – koristi se final |
| continue | Za preskakanje jedne iteracije |
| default | Dio switch-a , uvijek na kraju , uvijek se izvrši |
| do | Koristi se skupa sa while u do-while petlji |
| double | Primitivni tip -  double-precision 64-bit |
| else | Dio if-else , izvrši if -ako ne onda izvrši else |
| enum | Keyword za deklariranje enumerated (unchangeable) type, koristimo da varijable postavimo na predefinirane konstante |
| extends | Koristimo u nasljeđivanju  subclass (child) - the class that inherits from another class  superclass (parent) - the class being inherited from |
| final | Služi za kreiranje konstanti , da spriječimo nasljeđivanje i da spriječimo override metoda |
| finally | Koristimo skupa sa try , nakon try se izvrši finally blok |
| float | Primitivni tip - The float data type is a single-precision 32-bit |
| for | Za startanje for petlje |
| goto | Java nema goto al ima labele |
| if | Za testiranje uvjeta, ako je true if blok se izvrši |
| implements | Keyword za implementiranje interface-a |
| import |  |
| instanceof |  |
| int |  |
| interface |  |
| long |  |
| native |  |
| new |  |
| package |  |
| private |  |
| protected |  |
| public |  |
| return |  |
| short |  |
| static |  |
| super |  |
| switch |  |
| synchronized |  |
| this |  |
| throw |  |
| throws |  |
| transient |  |
| try |  |
| void |  |
| volatile |  |
| while |  |
| false |  |
| null |  |
| true |  |
| **SQL** | |
| **Općenito** | |
| table |  |
| column |  |
| view |  |
| index |  |
| constraint |  |
| stored procedure |  |
| trigger |  |
| create |  |
| drop |  |
| select |  |
| insert |  |
| update |  |
| delete |  |
| commit |  |
| inner/outer/left/right join - objasniti |  |
| **Vrste relacija između tablica** | |
| 1 – 1 |  |
| 1 – više |  |
| više - više |  |
| **SVN** | |
| **Općenito** | |
| Check-out |  |
| Commit |  |
| trunk |  |
| branch |  |
| **UNIX** | |
| **Osnovne naredbe** | |
| Ls |  |
| Cd |  |
| Mkdir |  |
| Grep |  |
| Cp |  |
| Rm |  |
| Mv |  |
| Rmdir |  |
| Touch |  |
| Chmod |  |
| Chown |  |
| Man |  |
| Ps |  |
| Kill |  |
| Kako izlistati stanje servisa |  |
| Gdje se definiraju mount pointi da budu perzistentni |  |
| Vrste distribucija |  |
| Koja je zadnja verzija referentnog kernela |  |
| **Specijalni direktoriji** | |
| / |  |
| . |  |
| .. |  |
| ~ |  |
| **Putanje** | |
| Apsolutna putanja |  |
| Relativna putanja |  |
| **Osnovni preglednici** | |
| Vrste osnovnih preglednika |  |
| Naredbe u „less“ pregledniku |  |
| **vi editor** | |
| Uređivanje teksta |  |
| Snimanje promjena |  |
| Brisanje retka |  |
| Pretraga teksta i navigacija (next/previous match) |  |
| **General know-how** | |
| SOAP |  |
| RegExp |  |
| Node-red |  |
| GMT / UTC / CEST |  |
| RabbitMQ |  |
| MosqitoMQ |  |
| Cassandra |  |
| NoSQL / NewSQL |  |
| ETSI |  |
| 3GPP |  |
| LTE |  |
| 5G |  |
| Što je podatak, a što informacija |  |
| Što je REGEXP, napisati regexp za IPv4 adresu i objasniti ga |  |
| Što je RFC |  |
| Kojim RFCom je opisan HTTP protokol (v 1.1), kojim HTTP2? |  |
| JSON |  |
| REST |  |
| WS |  |
| FYI |  |
| IIRC, IMHO |  |

# Praktični dio

Riješite navedene zadatke vodeći računa o točnosti rješenja, čitljivosti i razumljivosti koda. Svi java programi koje budete kreirali tijekom priprema trebaju biti dio istog Eclipse projekta pod nazivom PripremaPrakse\_<ImePrezime> (kreira se unutar zadatka 02). Neka rješenje svakog zadatka bude u zasebnom java paketu. Pakete nazovite ovako:

hr.atos.praksa.<imeprezime>.zadatak<broj zadatka>

gdje <imeprezime> predstavlja vaše ime i prezime, a <broj zadatka> dvoznamenkastu brojčanu oznaku zadatka (npr. 03 ili 13).

Primjer: rješenje petog zadatka studenta Ivana Horvata treba biti u java paketu:

hr.atos.praksa.ivanhorvat.zadatak05

## Zadatci

1. Instaliraj program Eclipse. Na stranici <http://www.eclipse.org/downloads/> nalazi se popis različitih alata iz Eclipse porodice. Za pripremu prakse bit će dovoljan Eclipse Standard. Odaberi verziju operacijskog sustava koji koristiš i preuzmi alat. Eclipse se ne instalira, nego jednostavno „raspakira“ na željenu lokaciju.
2. Kreiraj Java projekt i nazovi ga PripremaPrakse\_<ImePrezime>, pri čemu <ImePrezime> trebate zamijeniti svojim imenom i prezimenom (npr. PripremaPrakse\_IvanHorvat)
3. Kreiraj tzv. "Hello World" aplikaciju tj. aplikaciju koja će nakon pokretanja na konzolu ispisati riječi "Hello World, ovo je prva java aplikacija.".
4. Kreiraj program koji ima definirano polje pet različitih cjelobrojnih vrijednosti. Program treba provjeriti je li svaka od pet vrijednosti paran ili neparan broj i da li je višekratnik od jednog od idučih brojeva: 3, 5 ili 11, te ispisati poruku na konzolu u obliku „XY je paran broj“ ili „XY je neparan broj“ te „XY je višekratnik od 3 “ (ili 5 ili 11, zamijeniti broj u tekstu).
5. Kreiraj program koji ima definiran početak i kraj intervala te jedan brojač. Neka početak intervala bude manji od 10, a kraj intervala veći od 100. Program treba provjeriti jesu li početak i kraj intervala ispravno definirani te ispisati poruku o pogrešci ukoliko nisu. Ukoliko jesu program treba za svaki broj u zadanom intervalu napraviti sljedeće:
   * ukoliko je broj manji ili jednak 18, brojač treba povećati za 4
   * ukoliko je broj veći od 18, brojač treba umanjiti za 1
   * ukoliko je broj djeljiv s 20, treba prekinuti procesuiranje tog broja i prijeći na sljedeći
   * ukoliko je broj jednak ili veći od 75, treba prekinuti procesuiranje intervala te ispisati vrijednost brojača
6. Kreiraj program koji će koristeći for petlje automatizirati ispis tablice množenja u ovom obliku:

-------------------------------

: : : TABLICA MNOZENJA : : :

-------------------------------

\* | 1 2 3 4 5 6 7 8 9

-------------------------------

1 | 1 2 3 4 5 6 7 8 9

2 | 2 4 6 8 10 12 14 16 18

3 | 3 6 9 12 15 18 21 24 27

4 | 4 8 12 16 20 24 28 32 36

5 | 5 10 15 20 25 30 35 40 45

6 | 6 12 18 24 30 36 42 48 54

7 | 7 14 21 28 35 42 49 56 63

8 | 8 16 24 32 40 48 56 64 72

9 | 9 18 27 36 45 54 63 72 81

-------------------------------

: : : : : : : : :by Ime

-------------------------------

Umjesto "Ime" treba ispisati ime uneseno s konzole i pri tome pripaziti da zadnje slovo imena bude poravnato s desnim rubom tablice.

1. Kreiraj program koji će ispisati koliko ima brojeva djeljivih sa 6 u intervalu između neka dva prirodna broja unesena s konzole.
2. Kreiraj program koji će s konzole učitati redni broj mjeseca u nekoj prijestupnoj godini. Napraviti provjeru je li uneseni redni broj mjeseca ispravan (ne postoji npr. 15. mjesec), a zatim ispisati kalendar tog mjeseca u obliku:

P U S Č P S N

1 2 3 4 5 6 7

8 9 10 11 12 13 14

15 16 17 18 19 20 21

22 23 24 25 26 27 28

29 30 31

Pretpostavlja se kako prvi u mjesecu pada na ponedjeljak. Također, treba ispisati onoliko dana koliko ih taj mjesec po klasičnom kalendaru doista ima.

1. Luka je vaš dobar prijatelj. Nedavno vam je spomenuo kako bi volio imati grafičko rješenje prikaza njegovog troška struje njegove „krušne“ peći kroz godinu (on je naime pizza majstor i ima svoj restoran). Račun mu stalno varira, ali nikada ne prelazi 4500kn. Prvo ste mu rekli da još ne znate raditi grafički prikaz u Javi, ali vam je bilo žao ne ponuditi mu nikakvo rješenje pa ste na papiru nacrtali ovakav graf i rekli da znanje za napraviti ovakav prikaz imate:

5000kn - |

| x

4000kn - | x

|

3000kn - | x x

|

2000kn - | x

|

1000kn - | x x x

|

0kn - | x x

-- -- -- -- -- -- -- -- -- -- -- --

1 2 3 4 5 6 7 8 9 10 11 12

Također ste mu pojasnili kako će morati unijeti plaću za svaki pojedini mjesec, a onda će program iscrtati graf. Upozorili ste ga kako će vrijednosti plaće biti aproksimirane. Primjerice mjesečna plaća koja iznosi od 1501kn pa sve do 2500kn, na grafu će biti obilježena iksićem uz vrijednost 2000kn. On se naposljetku složio s vašim prijedlogom i od srca zahvalio. Stoga kreirajte program kojim ćete pomoći Luki.

1. Kreiraj java program koji će pretražiti sve datoteke s ekstenzijama "csv" i "txt" unutar jednog direktorija (foldera) kako bi provjerio nalazi li se u njima definirana tekstualna fraza. Nazive datoteka u kojima se fraza nalazi, ispisat će na konzolu. Putanju direktorija za pretraživanje kao i tekstualnu frazu treba unijeti s konzole.

Popis klasa koje bi mogle biti korisne:

java.io.File (za manipuliranje datotekama / direktorijima)

java.io.BufferedReader (za čitanje sadržaja datoteke)

java.lang.String (za manipulaciju sa tekstom)

Fraza usput može biti regular expression.

1. Kreiraj program koji će omogućiti interakciju s korisnikom na jednom od tri ponuđena jezika. Interakcija se treba sastojati od nekoliko međusobno izmijenjenih poruka. Prijevodi na pojedini jezik trebaju biti dostupni unutar java properties datoteka za svaki jezik odvojeno.
2. Napraviti program koji će omogućiti korisniku ispis statistike broja pojedinih riječi unutar datoteke koju odabere. Riječi trebaju biti poredane po abecedi i treba ignorirati velika/mala slova. Ispod se nalazi primjer ispisa:

U datoteci XX nalaze se sljedece rijeci:

------------------------

Rijec (broj ponavljanja)

------------------------

auto (17)

auta (2)

boje (19)

cipele (3)

je (39)

su (110)

trava (1)

zeleno (2)

------------------------

1. Kreiraj program koji se sastoji od jedne klase i sučelja koje klasa nasljeđuje. Sučelje treba definirati dvije različite metode po vlastitom izboru.
2. Kreiraj program kojim će korisnik izračunati površinu ispod neke krivulje opisane funkcijom y=f(x)=A\*funk(x)+B od neke početne do krajnje točke (T1,T2). Izračun izvesti numeričkom metodom. Ulazni parametri su A, B, T1i T2 te funkcija koja može biti sinus, kosinus, tangens ili kotangens. Pri rješavanju zadatka obavezno koristiti više klasa i(li) sučelja te nasljeđivanje.
3. Za jednu tvrtku potrebno je napraviti demo aplikaciju u kojoj će moći raditi administraciju zaposlenika te njihbovih radnih zadataka. Ovo su im najbitniji podatci:

Zaposlenici

ime

prezime

radno mjesto

oib

Zadaci

naziv

opis

osoba (1..n) koja radi na zadatku

tip (bug, task)

trenutni status (otvoren, zatvoren, u tijeku)

kompleksnost (broj)

potrošeno vrijeme (sati)

početni datum i vrijeme

završni datum i vrijeme

(napomena, početak i završetak ne znače nužno da je potrošeno vrijeme njihova razlika)

Potrebno je omogućiti administraciju na nekoliko razina tj. potrebno je definirati korisnička prava korisnika aplikacije po grupama: "admin", "superuser" i "user" tako da se omoguće sljedeće akcije:

* + kreiranje zadataka i zaposlenika (admin, superuser)
  + izlistanje zadataka i zaposlenika (admin, superuser, user)
  + izmjena zadataka i zaposlenika (admin)
  + brisanje zadataka i zaposlenika (admin)

Naposljetku aplikacija mora omogućiti kreiranje izvještaja:

* + po radnom mjestu tj. koliko radnika radi na pojedinom radnom mjestu
  + utrošeno vrijeme po osobi
  + najduže otvoren zadatak

Izvještaje mogu kreirati samo korisnici s admin ili superuser pravima.

Sučelje za komunikaciju s korisnicima neka bude konzola, a podaci se trebaju spremati u tekstualne datoteke ili bazu podataka po izboru.

# Upute za završni izvještaj

Završni izvještaj sadrži rezultat svladavanja teorijske i praktične grupe zadataka:

* Završni izvještaj teorijskog dijela podrazumijeva da u drugi stupac tablice iz poglavlja 1. Teorijski dio unesete vaše razumijevanje pojedinog pojma navedenog u prvom stupcu te tablice. Preporuča se pojam opisivati vlastitim riječima na hrvatskom jeziku.
* Za završni izvještaj praktičnog dijela trebate napraviti eksport Eclipse projekta PripremaPrakse\_<ImePrezime> (koji sadrži java programe tj. rješenja svih zadataka koje ste riješili tijekom priprema) u zip datoteku jednakog naziva (PripremaPrakse\_<ImePrezime>.zip). Tako kreiranu zip datoteku trebate umetnuti na kraj ovog word dokumenta.

Nakon što ste odradili navedena dva koraka spremite ovaj word dokument. Prije nego ga pošaljete provjerite jeste li promijenili ImePrezime iz naziva dokumenta u vaše ime i prezime.