MR ZORAN S. VELIČKOVIĆ, DIPL. INŽ.

INTERNET PROGRAMIRANJE

PRAKTIKUM LABORATORIJSKIH VEŽBI

Java

NIŠ, 2008.

Broj poena

_____/_____/_____	_____
S T U D E N T / Br. Indeksa / Grupa	POTVRĐUJE

NAPOMENE

PREDGOVOR

Praktikum za obavljanje laboratorijskih vežbi iz predmeta “Internet programiranje ” je namenjen studentima Elektrotehničkog odseka Visoke tehničke škole strukovnih studija u Nišu. Osnovni cilj laboratorijskih vežbi je da se teorijska poglavlja upotpune praktičnim znanjima i veštinama. Iz ovog razloga laboratorijske vežbe predstavljaju OBAVEZNI deo nastave.

Laboratorijske vežbe su prilagođene raspoloživim tehničkim resursima i pokrivaju sva važnija teorijska poglavlja. Koncepcija ovog Praktikuma je da se studenti postepeno uvode u sve složenije programerske zadatke uz istovremeno davanje korisnih uputstava i saveta pri praktičnim realizacijama. Da bi se postigao maksimalni učinak ovih vežbi, neophodna je teorijska priprema studenata.

Vođenje tehničkog izveštaja o izvršenim vežbama je jednako značajan zadatak kao i sama realizacija vežbi. Ovaj Praktikum usmerava studente na tehnički prihvatljiv način prikaza toka podataka kao i prikaz izlaznih rezultata. Posle završene vežbe studenti treba da daju odgovore na postavljena pitanja i zapišu svoja zapažanja na za to predviđenom mestu. Preduslov započinjanja naredne laboratorijske vežbe je kompletiranje tehničkog izveštaja prethodne.

Osnovno vežbanje se obavlja u okruženju JDK-a, uglavnom sa komandne linije. Pojedine vežbe su predviđene da koriste integrisano razvojno okruženje Eclipse. Korišćenjem ova dva razvojna paketa za Javu savladavaju se osnove jezika i stiče neophodno iskustvo za rad u profesionalnom radnom okruženju.

Student ne sme početi vežbu pre nego što za to dobije dozvolu. U toku izrade laboratorijskih vežbi studenti moraju poštovati pravila o radu u laboratoriji kako bi se izbegle situacije koje mogu naneti štetu bilo studentima, bilo opremi.

Posle završenih laboratorijskih vežbi studenti polažu završni kolokvijum koji se odnosi na odbranu predatih izveštaja. Preduslov za polaganje ispita “Internet programiranje ” jesu OBAVLJENE i OVERENE Laboratorijske vežbe od strane Predmetnog nastavnika ili asistenta, čime se potvrđuje da je student uspešno obavio predviđeni praktični deo nastave.

Niš, Oktobar 2008. Autor

SADRŽAJ

VEŽBA BR. 1.

OSNOVE RADA U JAVA DEVELOPMENT KIT-u (JDK)

CILJ VEŽBE: Upozanavanje sa osnovama rada Java Development Kit-u (JDK). Postavljanje environment promenljivih u JDK-u i setovanje odgovarajućih parametara. Realizacija jednostavnih konzolskih programa.

ZADATAK 1: Sa SUN-ovog sajta www.sum.com preuzeti poslednju verziju JDK-a. Obratite pažnju na verziju operativnog sistema. Navesti imena verzija datoteka koje se nude za preuzimanje i operativne sisteme koje SUN podržava:

_____
_____
_____
_____
_____

Preuzeta je verzija _____ čija je veličina _____.

Kakvog je tipa preuzeto radno okruženje?

_____.

Objasniti ulogu i značaj programskog paketa JRE.

_____
_____
_____
_____
_____

ZADATAK 2: Instalirati preuzet programski paket na target računaru. Postaviti odgovarajuće vrednosti sistemskih promenljivih kako bi se prevođenje i izvršenje programa moglo vršiti iz bilo kojeg foldera.

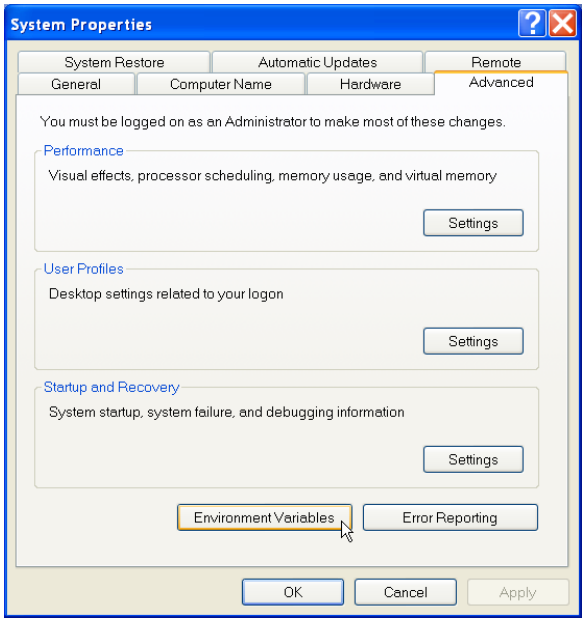
Po instaliranju programski paketa JDK i JRE u **System properties** kliknuti na Environment variable. Promenljivu **path** postaviti na korektnu vrednost. Možete koristiti pridružen editor da obavite ovaj posao.

Nabrojati i objasniti osnovne faze u izradi programa u razvojnom okruženju JDK-a.

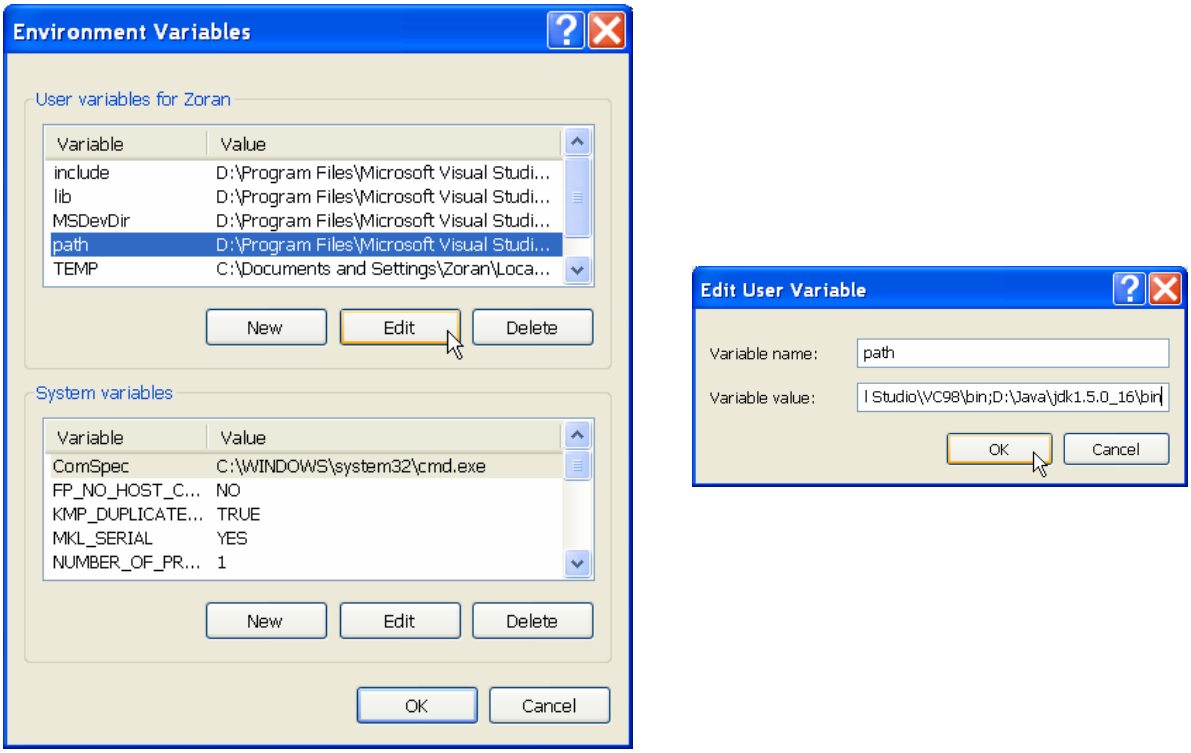
_____
_____
_____
_____

Objasniti pojam JVM-a.

_____
_____
_____
_____
_____



Slika 1.1



Slika 1.2

Upisati i objasniti vrednosti promenljivih vezanih za razvojno okruženje JDK-a.

ZADATAK 3: U tekst editoru uneti jednostatan program napisan u Javi za ispis odgovarajućeg teksta u komandnom prozoru. Zapamtite izvorni kod pod imenom: Example.java. U rezervisanom prostoru ispod je prikazan izvorni kod ovog programa.

```
/*
   This is a simple Java program.
   Call this file "Example.java".
*/
class Example {
  // Your program begins with a call to main().
  public static void main(String args[]) {
    System.out.println("This is a simple Java program.");
  }
}
```

ZADATAK 3: Komentarisati svaki red prikazanog programskog koda.

```
/*
   This is a simple Java progr
*/
```

_____
_____

```
class Example {
```

_____
_____
_____

```
// Your program begins with a call to main().
```

_____
_____
_____

```
public static void main(String args[]) {
```

_____
_____
_____

```
System.out.println("This is a simple Java program.");
```

_____
_____
_____

Napisati komandu za kompajliranje ovog izvornog koda.

_____
_____

Navesti raspoložive opcije Java kompajlera i objasnite najvažnije.

Šta je rezultat rada Java kompajlera? Koje su ekstenzije dobijenih fajlova?

Napisati komandu za startovanje ovog programa.

ZADATAK 4: Izmeniti izvorni kod u prethodnom primeru da bi se štampalo Vaše ime i prezime u jednom redu, dok bi se u drugom redu štampao Vaš studentski status.

ZADATAK 5: Šta je izlazni rezultat sledećeg izvornog Java koda? U čemu je razlika između ova dva primera?

```
class Example2 {
  public static void main(String args[]) {
    int num; // this declares a variable called num

    num = 100; // this assigns num the value 100
    System.out.println("This is num: " + num);

    num = num * 2;

    System.out.print("The value of num * 2 is ");
    System.out.println(num);
  }
}
```

ZAKLJUČAK:

U Nišu

POTVRĐUJE

VEŽBA BR. 2.

TIPOVI PODATAKA, OPERATORI I UPRAVLJAČKE NAREDBE

CILJ VEŽBE: Upozanavanje sa osnovnim tipovima podataka u Javi. Korišćenje operatora i upravljačkih naredbi za realizaciju programskih zahteva.

ZADATAK 1: Navesti osnovne tipove podataka u Javi i objasniti njihove osnovne karakteristike.

byte_____
short_____
int_____
long_____
double_____

ZADATAK 2: Koristeći promenljive tipa float ili double izračunati površinu kruga. Izvorni kod koji je dat u nastavku teksta uneti pomuću tekst editora i zapamti pod imenom_____.

```
// Compute the area of a circle.
class Area {
  public static void main(String args[]) {
    double pi, r, a;

    r = 10.8; // radius of circle
    pi = 3.1416; // pi, approximately
    a = pi * r * r; // compute area

    System.out.println("Area of circle is " + a);
  }
}
```

Kompajler je formirao fajl pod nazivom: _____.

Posle startovanja programa na komandnoj liniji se prikazuje:

_____.

ZADATAK 3: Koristeći promenljivu tipa char predstaviti karakter X. Tretirajući promenljivu tipa char kao integer formirati karakter Y. Odštampati ovako dobijene karaktere.

```
// char variables behave like integers.
class CharDemo2 {
  public static void main(String args[]) {
    char ch1;

    ch1 = 'X';
    System.out.println("ch1 contains " + ch1);
    ch1++; // increment ch1
    System.out.println("ch1 is now " + ch1);
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:
_____.

ZADATAK 4: Formirajte niz koji se sastoji od brojeva dana u svakom mesecu. Izvorni kod koji je dat u nastavku teksta uneti pomuću tekst editora i zapamti pod imenom_____.

```
class Array {
  public static void main(String args[]) {
    int month_days[];
    month_days = new int[12];
    month_days[0] = 31;
    month_days[1] = 28;
    month_days[2] = 31;
    month_days[3] = 30;
    month_days[4] = 31;
    month_days[5] = 30;
    month_days[6] = 31;
    month_days[7] = 31;
    month_days[8] = 30;
    month_days[9] = 31;
    month_days[10] = 30;
    month_days[11] = 31;
    System.out.println("April has " + month_days[3] + "
days.");
  }
}
```

Kompajler je formirao fajl pod nazivom: _____.

Posle startovanja programa na komandnoj liniji se prikazuje:
_____.

ZADATAK 5: Izvorni kod grananja programa if-else-if naredbom je dat u nastavku.

```
class IfElse {
  public static void main(String args[]) {
    int month = 4; // April
    String season;
    if(month == 12 || month == 1 || month == 2)
      season = "Winter";
    else if(month == 3 || month == 4 || month == 5)
      season = "Spring";
    else if(month == 6 || month == 7 || month == 8)
      season = "Summer";
    else if(month == 9 || month == 10 || month == 11)
      season = "Autumn";
    else
      season = "Bogus Month";
    System.out.println("April is in the " + season + ".");
  }
}
```

Kompajler je formirao fajl pod nazivom: _____.

Posle startovanja programa na komandnoj liniji se prikazuje:
_____.

ZADATAK 6: Izvorni kod grananja programa `switch` naredbom je dat u nastavku. Pod kojim imenom treba zapamtiti izvorni kod? _____.

```
// A simple example of the switch.
class SampleSwitch {
  public static void main(String args[]) {
    for(int i=0; i<6; i++)
      switch(i) {
        case 0:
          System.out.println("i is zero.");
          break;
        case 1:
          System.out.println("i is one.");
          break;
        case 2:
          System.out.println("i is two.");
          break;
        case 3:
          System.out.println("i is three.");
          break;
        default:
          System.out.println("i is greater than 3.");
      }
  }
}
```

Kompajler je formirao fajl pod nazivom: _____.

Posle startovanja programa na komandnoj liniji se prikazuje:

ZAKLJUČAK:

_____
_____
_____
_____

U Nišu

POTVRĐUJE

_____ _____

VEŽBA BR. 3.

OSNOVE KLASA U JAVI

CILJ VEŽBE: Upozanavanje sa osnovama OOP-a. Formiranje jednostavnih klasa i parametarskih konstruktora. Preklapanje konstruktora. Preklapanje metoda.

ZADATAK 1: Formirati klasu `Box` koja sadrži tri promenljive: `double width`, `double height` i `double depth`. Napravite dva objekat tipa `Box` korišćenjem operatora `new`. Izračunti zapremine formiranih objekata. Izvorni kod klase i primer koji je koristi dat je u nastavku.

```
// This program declares two Box objects.

class Box {
  double width;
  double height;
  double depth;
}
class BoxDemo2 {
  public static void main(String args[]) {
    Box mybox1 = new Box();
    Box mybox2 = new Box();
    double vol;

// assign values to mybox1's instance variables
    mybox1.width = 10;
    mybox1.height = 20;
    mybox1.depth = 15;

/* assign different values to mybox2's instance variables */
    mybox2.width = 3;
    mybox2.height = 6;
    mybox2.depth = 9;

// compute volume of first box
    vol = mybox1.width * mybox1.height * mybox1.depth;
    System.out.println("Volume is " + vol);

// compute volume of second box
    vol = mybox2.width * mybox2.height * mybox2.depth;
    System.out.println("Volume is " + vol);
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Šta su to instance klase? Koje instance imamo u prethodnom primeru?

_____
_____
_____

Opišite upotrebu operatora `new` i način pristupa promenljivama instance.

_____
_____
_____
_____

ZADATAK 2: Dodati klasi `Box` metodu `volume()`.

```
// This program includes a method inside the box class.

class Box {
  double width;
  double height;
  double depth;

  // display volume of a box
  void volume() {
    System.out.print("Volume is ");
    System.out.println(width * height * depth);
  }
}

class BoxDemo3 {
  public static void main(String args[]) {
    Box mybox1 = new Box();
    Box mybox2 = new Box();

    // assign values to mybox1's instance variables
    mybox1.width = 10;
    mybox1.height = 20;
    mybox1.depth = 15;

    /* assign different values to mybox2's
       instance variables */
    mybox2.width = 3;
    mybox2.height = 6;
    mybox2.depth = 9;

    // display volume of first box
    mybox1.volume();

    // display volume of second box
    mybox2.volume();
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Opišite osnovne razlike u programskom kodu kod zadatka 1 i 2.

ZADATAK 3: Formirajte parametarski konstruktor klase `Box` koji će vrednosti parametra kutije inicijalizovati prilikom poziva odgovarajućeg konstruktora. Nova verzija klase `Box` je data u nastavku.

```
/* Here, Box uses a parameterized constructor to
   initialize the dimensions of a box.
*/
class Box {
  double width;
  double height;
  double depth;

  // This is the constructor for Box.
  Box(double w, double h, double d) {
    width = w;
    height = h;
    depth = d;
  }

  // compute and return volume
  double volume() {
    return width * height * depth;
  }
}
```

Šta je karakteristično za klasu `Box` iz datog primera? Kako se naziva metoda `Box(double w, double h, double d)` a kako `volume()` iz prethodnog primera?

Opišite ulogu rezervisane reči `return` u prethodnom primeru.

Koji se tip promenljive vraća prilikom poziva metode `volume()`? Zašto?

Izvorni kod klase koja koristi ovako formiranu klasu Box je dat u nastavku.

```
class BoxDemo7 {
  public static void main(String args[]) {
    // declare, allocate, and initialize Box objects
    Box mybox1 = new Box(10, 20, 15);
    Box mybox2 = new Box(3, 6, 9);

    double vol;

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume is " + vol);
  }

}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasnite pojam automatske konverzije tipova podataka u Javi.

Objasnite pojam izričite konverzije tipova podataka u Javi.

ZAKLJUČAK:

U Nišu

POTVRĐUJE

VEŽBA BR. 4.

PREKLAPANJE METODA I KONSTRUKTORA

CILJ VEŽBE: Upozanavanje sa naprednim pojmovima klasa. Usvojiti i primeniti pojmove preklapanje metoda i konstruktora.

ZADATAK 1: Formirati preklopljenu metodu `test()` koja koristi različiti broj parametara za razlikovanje dve verzije metode. Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu.

```
// Demonstrate method overloading.
class OverloadDemo {
  void test() {
    System.out.println("No parameters");
  }
  // Overload test for one integer parameter.
  void test(int a) {
    System.out.println("a: " + a);
  }
  // Overload test for two integer parameters.
  void test(int a, int b) {
    System.out.println("a and b: " + a + " " + b);
  }
  // overload test for a double parameter
  double test(double a) {
    System.out.println("double a: " + a);
    return a*a;
  }
}
```

Klasa koja koristi preklopljene metode test() je data u nastavku.

```
class Overload {
  public static void main(String args[]) {
    OverloadDemo ob = new OverloadDemo();
    double result;

    // call all versions of test()
    ob.test();
    ob.test(10);
    ob.test(10, 20);
    result = ob.test(123.2);
    System.out.println("Result of ob.test(123.2): " + result);
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Koliko je puta preklopljena metoda `test()`? Po čemu se razlikuje četvrta metoda `test()`?

_____
_____
_____

Šta se dešava sa pozivom preklopljene metode ako se tip parametara ne slaže u potpunosti? Kakvu ulogu igra Javina automatska konverzija tipova? Da li se mogu preklapati konstruktori? Da li se mogu koristiti objekti kao parametri u pozivnim funkcijama?

_____
_____
_____
_____

ZADATAK 2: Formirati preklopljeni konstruktor klase Box koji će rešiti specifične tipove kutija. Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu.

```
/* U ovoj vežbi Box definiše tri konstruktora kako bi se
inicijalizovalle dimenzije Box-a na različite načine
*/
class Box {
  double width;
  double height;
  double depth;

  // constructor used when all dimensions specified
  Box(double w, double h, double d) {
    width = w;
    height = h;
    depth = d;
  }

  // constructor used when no dimensions specified
  Box() {
    width = -1;  // use -1 to indicate
    height = -1; // an uninitialized
    depth = -1;  // box
  }

  // constructor used when cube is created
  Box(double len) {
    width = height = depth = len;
  }

  // compute and return volume
  double volume() {
    return width * height * depth;
  }
}
```

Klasa koja koristi preklopljene konstruktore `Box()` je data u nastavku.

```
class OverloadCons {
  public static void main(String args[]) {
    // create boxes using the various constructors
    Box mybox1 = new Box(10, 20, 15);
    Box mybox2 = new Box();
    Box mycube = new Box(7);

    double vol;

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume of mybox1 is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume of mybox2 is " + vol);

    // get volume of cube
    vol = mycube.volume();
    System.out.println("Volume of mycube is " + vol);
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasniti pojam polimorfizma kod Jave. Kakva je veza između polimorfizma i preklapanja konstruktora odnosno metoda?

_____
_____
_____
_____

ZAKLJUČAK:

_____
_____
_____
_____

U Nišu

POTVRĐUJE

_____

_____

VEŽBA BR. 5.

NASLEĐIVANJE KLASA, REDEFINISANJE METODA

CILJ VEŽBE: Upozanavanje sa naprednim pojmovima klasa. Usvojiti i primeniti pojmove nasleđivanje klasa i redefinisanja metoda.

ZADATAK 1: Formirati natklasu A i potklasu B. Za nasleđivanje klasa koristiti rezervisanu reč `extends`. Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu.

```
// Jednostavan primer nasleđivanja (inheritance).

// Kreiranje natklase (superclass).
class A {
  int i, j;

  void showij() {
    System.out.println("i and j: " + i + " " + j);
  }
}

// Kreiranje podklase (suberclass) nasleđivanjem clase A.
class B extends A {
  int k;

  void showk() {
    System.out.println("k: " + k);
  }
  void sum() {
    System.out.println("i+j+k: " + (i+j+k));
  }
}
```

Klasa koja koristi nasleđene klase je data u nastavku.

```
class SimpleInheritance {
  public static void main(String args[]) {
    A superOb = new A();
    B subOb = new B();

    // The superclass may be used by itself.
    superOb.i = 10;
    superOb.j = 20;
    System.out.println("Contents of superOb: ");
    superOb.showij();
    System.out.println();

    /* The subclass has access to all public members of
       its superclass. */
    subOb.i = 7;
    subOb.j = 8;
    subOb.k = 9;
```

```
    System.out.println("Contents of subOb: ");
    subOb.showij();
    subOb.showk();
    System.out.println();

    System.out.println("Sum of i, j and k in subOb:");
    subOb.sum();
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasniti dobijene rezultate. Da li je korišćenje klase B nezavisno od klase A?

Da li potklasa B može biti istovremeno natklasa neke druge klase (recimo C)? Zašto?

Objasniti ulogu rezervisane reči `super` i `this`.

Kada se smatra da je metoda redefinisana?

Od čega zavisi koja će se verzija redefinisane metode pozvati na izvršenje? Objasnite pojam dinamičkog razrešavanja metoda pri izvršenju. Kakva je uloga JVM-a?

Objasniti pojam apstraktnih klasa.

ZADATAK 2: Napisati program koji pravi natklasu `Figure` koja sadrži dimenzije 2D objekata. Klasa definiše metodu `area()`, koja izračunava površinu objekta. Izvedite iz klase `Figure` dve potklase `Rectangle` i `Triangle`. Svaka od ovih klasa treba da redefiniše metodu `area()`, tako da vraća korektnu vrednost figure.

```
// Using run-time polymorphism.
class Figure {
  double dim1;
  double dim2;

  Figure(double a, double b) {
    dim1 = a;
    dim2 = b;
  }

  double area() {
    System.out.println("Area for Figure is undefined.");
    return 0;
  }
}

class Rectangle extends Figure {
  Rectangle(double a, double b) {
    super(a, b);
  }

  // override area for rectangle
  double area() {
    System.out.println("Inside Area for Rectangle.");
    return dim1 * dim2;
  }
}

class Triangle extends Figure {
  Triangle(double a, double b) {
    super(a, b);
  }

  // override area for right triangle
  double area() {
    System.out.println("Inside Area for Triangle.");
    return dim1 * dim2 / 2;
  }
}
```

Klasa koja koristi redefinisane metode je data u nastavku.

```
class FindAreas {
  public static void main(String args[]) {
    Figure f = new Figure(10, 10);
    Rectangle r = new Rectangle(9, 5);
    Triangle t = new Triangle(10, 8);

    Figure figref;

    figref = r;
    System.out.println("Area is " + figref.area());

    figref = t;
    System.out.println("Area is " + figref.area());

    figref = f;
    System.out.println("Area is " + figref.area());
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasniti prednosti mehanizma nasleđivanja i polimorfizma kod Jave.

Kako se pristupa izračunavanju površine kod različitih geometrijskih oblika? U čemu je razlika kod ovakvog pristupa u odnosu na C?

Objasniti uticaj modifikatora `final` pri nasleđivanju klasa kod Jave.

U Nišu

POTVRĐUJE

VEŽBA BR. 6.

OBRADA IZUZETAKA

CILJ VEŽBE: Upozanavanje sa programiranjem baziranom na mehanizmu izuzetaka (exception). Izuzetak u ovom smislu predstavlja grešku pri izvršenju programa. Usvojiti i primeniti strukturu tray-catch. Upoznati se sa klasom `Throwable` i njenim potklasama `Exception` i `Error`.

ZADATAK 1: Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu. Kompalirajte i izvršite program. Primetite da će `metoda()` izazvati grešku prilikom pokušaja deljenja nulom. Ovaj događaj se naziva izuzetkom.

```
class Exc1 {
  static void subroutine() {
    int d = 0;
    int a = 10 / d;
  }
  public static void main(String args[]) {
    Exc1.subroutine();
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasnite značenje ispisanih poruka. Šta se dešava sa izvršenjem programa?

_____

_____

_____

ZADATAK 2: Napisati programski kod koji će obraditi nastalu grešku. Iskoristiti izuzetak tipa `ArithmeticExcception` i programsku strukturu `tray-catch`.

```
class Exc2 {
  public static void main(String args[]) {
    int d, a;

    try { // monitor a block of code.
      d = 0;
      a = 42 / d;
      System.out.println("This will not be printed.");
    } catch (ArithmeticException e) { // catch divide-by-zero
error
      System.out.println("Division by zero.");
    }
    System.out.println("After catch statement.");
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasniti dobijene rezultate. Šta se postiže obradom izuzetaka? Da li se više struktura `try-catch` može ugnezditi?

ZADATAK 3: Napisati program koji "hvata" više tipova izuzetaka u jednom `tary` bloku. Koristiti izuzetke tipa `ArithmeticExceptions` i `ArrayIndexOutOfBoundsException`. Program startovati sa i bez argumenata sa komandne linije.

```
// Demonstrate multiple catch statements.
class MultiCatch {
  public static void main(String args[]) {
    try {
      int a = args.length;
      System.out.println("a = " + a);
      int b = 42 / a;
      int c[] = { 1 };
      c[42] = 99;
    } catch(ArithmeticException e) {
      System.out.println("Divide by 0: " + e);
    } catch(ArrayIndexOutOfBoundsException e) {
      System.out.println("Array index oob: " + e);
    }
    System.out.println("After try/catch blocks.");
  }
}
```

Posle startovanja programa na komandnoj liniji bez argumenata se prikazuje:

Posle startovanja programa na komandnoj sa argumenaom se prikazuje:

Objasniti upotrebljene tipove izuzetaka.

Objasniti dobijene rezultate.

ZADATAK 4: Napravite, "bacite" i "uhvatite "izuzetak `NullPointerException`. Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu. Kompalirajte i izvršite program.

```
 / Demonstrate throw.
class ThrowDemo {
  static void demoproc() {
    try {
      throw new NullPointerException("demo");
    } catch(NullPointerException e) {
      System.out.println("Uhvacen unutar metode.");
      throw e; // ponovno bacanje izuzetka
    }
  }

  public static void main(String args[]) {
    try {
      demoproc();
    } catch(NullPointerException e) {
      System.out.println("Ponovo uhvacen: " + e);
    }
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasniti dobijene rezultate.

_____
_____
_____

Objasniti programski iskaz: `throw new NullPointerException("demo").`

_____
_____
_____

Objasniti programski iskaz: `throw e`

_____
_____
_____

Šta se štampa u okviru naredbe `println` iz objekta `e` napravljenog prilikom generisanja izuzetaka.

_____
_____
_____

U Nišu

_____

POTVRĐUJE

_____

VEŽBA BR. 7.

VIŠENITNO PROGRAMIRANJE

CILJ VEŽBE: Upozanavanje sa pojmom višenitnog programiranja (`multithread`). Usvojiti pojmove sinhronizacije izvršavanja niti. Upoznati se sa klasom `Thread` i interfejsom `Runnable`. Razmotriti upotrebu sledećih metoda iz klase `Thread`: `getName(),run(),sleep(),start(), currentThred()`.

ZADATAK 1: Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu. Kompalirajte i izvršite program.

```
// Controlling the main Thread.
class CurrentThreadDemo {
  public static void main(String args[]) {
    Thread t = Thread.currentThread();

    System.out.println("Current thread: " + t);

    // change the name of the thread
    t.setName("My Thread");
    System.out.println("After name change: " + t);

    try {
      for(int n = 5; n > 0; n--) {
        System.out.println(n);
        Thread.sleep(1000);
      }
    } catch (InterruptedException e) {
      System.out.println("Main thread interrupted");
    }
  }
}
```

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasnite značenje ispisanih poruka.

_____

_____

_____

Objasniti programski iskaz: `Thread t = Thread.currentThread().`

_____

_____

_____

Objasniti programski iskaz: `t.setName("My Thread");`

_____

_____

_____

Objasniti programski iskaz: `Thread.sleep(1000);`

_____

_____

ZADATAK 2: Napraviti programsku nit realizacijom interfejsa `Runnable`. Klasa koja realizuje interfes `Runnable` treba da ima realizovanu metodu `run()`.

```
// Pravljenje druge niti.
class NewThread implements Runnable {
  Thread t;

  NewThread() {
    // Pravljenje nove, druge niti
    t = new Thread(this, "Demo NIT");
    System.out.println("Nit potomak: " + t);
    t.start(); // Startovanje izvršavanja niti
  }

  //Ulazna tačka u drugu nit.
  public void run() {
    try {
      for(int i = 5; i > 0; i--) {
        System.out.println("Nit potomak: " + i);
        Thread.sleep(500);
      }
    } catch (InterruptedException e) {
      System.out.println("Nit potomak prekinuta.");
    }
    System.out.println("Izlaz iz niti potomak.");
  }
}

class ThreadDemo {
  public static void main(String args[]) {
    new NewThread(); // pravljenje nove niti

    try {
      for(int i = 5; i > 0; i--) {
        System.out.println("Glavna nit: " + i);
        Thread.sleep(1000);
      }
    } catch (InterruptedException e) {
      System.out.println("Glavna nit je prekinuta.");
    }
    System.out.println("Izlazak iz glavne niti.");
  }
}
```

Objasniti programski iskaz: `t = new Thread(this, "Demo NIT")`

Posle startovanja programa na komandnoj liniji se prikazuje:

Objasniti dobijene rezultate.

Objasniti u kom delu programa se programske niti izvršavaju uporedo.

Objasniti razliku između višenitnog i multitasking načina programiranja.

Koja su ograničenja višenitnog načina programiranja?

U Nišu

POTVRĐUJE

VEŽBA BR. 8.

OSNOVE APLETA

CILJ VEŽBE: Upozanavanje sa pojmom Apleta. Korišćenje apleta u HTML-u. Radno okruženje za razvoj apleta i izvršni Javin sistem.

ZADATAK 1: Unesite dat izvorni kod u tekst editoru i dajte odgovarajuće ime izvornom fajlu. Kompalirajte program.

```
import java.awt.*;
import java.applet.*;
/*
<applet code="SimpleApplet" width=200 height=60>
</applet>
*/

public class SimpleApplet extends Applet {
  public void paint(Graphics g) {
    g.drawString("A Simple Applet", 20, 20);
  }
}
```

Pod kojim imenom treba zapamtiti izvorni kod ovog apleta?

_____

Pod kojim imenom treba očekivati kompajlirai fajl.

_____

Objasniti programski iskaz: `import java.awt.*;import java.applet.*;`

_____

_____

_____

Objasniti programski iskaz:

```
/*
<applet code="SimpleApplet" width=200 height=60>
</applet>
*/
```

_____

_____

_____

Objasniti parametar metode `paint()`:

_____

_____

_____

Objasniti parametar metode `drawString()`:

_____

_____

Kako se može startovati izvršenje ovog apleta?

_____

_____

_____

ZADATAK 2: Zadati boju pozadine i prednjeg plana apleta. Istovremeno prikazivati imena funkcija kako se pozivaju.

```
 /* A simple applet that sets the foreground and
   background colors and outputs a string. */

import java.awt.*;
import java.applet.*;
/*
<applet code="Sample" width=300 height=50>
</applet>
*/

public class Sample extends Applet{
  String msg;

  // set the foreground and background colors.
  public void init() {
    setBackground(Color.cyan);
    setForeground(Color.red);
    msg = "Inside init( ) --";
  }

  // Initialize the string to be displayed.
  public void start() {
    msg += " Inside start( ) --";
  }

  // Display msg in applet window.
  public void paint(Graphics g) {
    msg += " Inside paint( ).";
    g.drawString(msg, 10, 30);
  }
}
```

Objasniti programski iskaz: `setBackground(Color.cyan);`

_____
_____
_____

Objasniti programski iskaz: `msg = "Inside init( ) --";`

_____
_____
_____

Objasniti programski iskaz:

```
   public void paint(Graphics g) {
    msg += " Inside paint( ).";
    g.drawString(msg, 10, 30);
  }
```

_____
_____
_____

ZADATAK 3: Iz HTML koda proslediti parametre apletu koji definišu tip slova, veličinu slova i razmak.

```
import java.awt.*;
import java.applet.*;
/*
<applet code="ParamDemo" width=300 height=80>
<param name=fontName value=Courier>
<param name=fontSize value=14>
<param name=leading value=2>
<param name=accountEnabled value=true>
</applet>
*/

public class ParamDemo extends Applet{
  String fontName;
  int fontSize;
  float leading;
  boolean active;

  // Initialize the string to be displayed.
  public void start() {
    String param;

    fontName = getParameter("fontName");
    if(fontName == null)
      fontName = "Not Found";

    param = getParameter("fontSize");
    try {
      if(param != null) // if not found
        fontSize = Integer.parseInt(param);
      else
        fontSize = 0;
    } catch(NumberFormatException e) {
      fontSize = -1;
    }

    param = getParameter("leading");
    try {
      if(param != null) // if not found
        leading = Float.valueOf(param).floatValue();
      else
        leading = 0;
    } catch(NumberFormatException e) {
      leading = -1;
    }

    param = getParameter("accountEnabled");
    if(param != null)
      active = Boolean.valueOf(param).booleanValue();
  }
  // Display parameters.
  public void paint(Graphics g) {
    g.drawString("Font name: " + fontName, 0, 10);
    g.drawString("Font size: " + fontSize, 0, 26);
    g.drawString("Leading: " + leading, 0, 42);
    g.drawString("Account Active: " + active, 0, 58);
  }
}
```

Objasniti način prenosa parametara na konkretnom primeru iz datog koda:

_____

_____

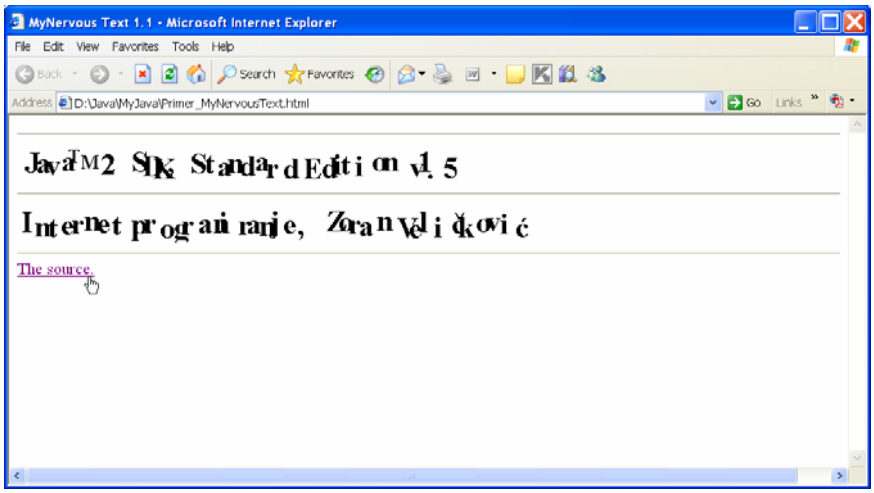
_____

ZADATAK 4: Sa sajta `sun.com` učitati izvorni kod apleta `NervousText.java`. Proučiti dat izvorni kod i način prenosa parametara u aplet. Proučiti strukturu apleta i dati svoja zapažanja.

_____

_____

_____



ZADATAK 5: Napisati HTML kod kako bi se dobio izgled stranice kao na slici:

U Nišu

POTVRĐUJE

_____

_____