Písomka 29. 11. 2007 A Meno:

I. V tejto otázke predpokladáme stručné a jasne formulované odpovede. Maximálne tri vety na každú otázku. Každá správna odpoveď je za 1 bod. Celá otázka je za 10 bodov.

1. Čo je pole? Z čoho sa pole skladá?
2. Čo je to referencia? Ako deklarujeme referenciu na pole?
3. Ako sa vytvára pole?
4. Ako je možné pole inicializovať?
5. Predpokladajte, že ste vytvorili pole, ktoré má 100 prvkov. Aký interval indexov je možné použiť, ak sa chceme odvolávať na prvky tohto poľa?
6. Určte, ktoré z nasledujúcich výrazov Javy sú spravné. U každého výrazu určte, čo sa bude v programe robiť:

a)
```
double arr[]; arr=new double[10];
```
b)
```
double aaa[]; aaa=new int[100];
```
c)
```
double bbb[]; bbb[]={1., 2., 3., 4., 5., 6.};
```
d)
```
double aaa[] = {1., 2., 3., 4., 5.};
for (i=1; i<aaa.length; i++)
   System.out.println("i = " + aaa[i]);
```
e)
```
double aaa[] = {1., 2., 3., 4., 5.};
for (i=aaa.length-1; i>=0; i--)
   System.out.println("i = " + aaa[i]);
```

II. 10 bodov
Implementujte nasledovnú metódu:

Premixovaný reťazec

Metóda: `public String premixuj(String s)`

Opis: Vráti reťazec, ktorý vznikne zlepením 2 reťazcov: prvý reťazec pozostáva z písmen na párnych pozíciách reťazca *s* a druhý reťazec z písmen na nepárnych pozíciách.
`premixuj("Programovanie") -> "Pormvnergaoai"`

III. 8 bodov
Implementujte nasledovnú metódu:

Najvzdialenejšia dvojica

Metóda: `public double najvzdialenejsie(Turtle[] korytnacky)`

Opis: Vráti vzdialenosť medzi 2 najvzdialenejšími korytnačkami v poli *korytnacky*.
V poli *korytnacky* je každá korytnačka práve raz, žiaden prvok poľa nie je *null* a pole má dĺžku aspoň 5.

Niektoré užitočné metódy objektov triedy *Turtle*	
`double getX()`	Vráti x-ovú súradnicu aktuálnej pozície korytnačky.
`double getY()`	Vráti y-ovú súradnicu aktuálnej pozície korytnačky.
`double distanceTo(double x, double y)`	Vráti vzdialenosť korytnačky k bodu [x, y]

IV. Uvažujme nasledujúcu triedu: (12 bodov)

```
public class Vektor {
      public double x;
      public double y;

      public Vektor(double x, double y) {
            this.x = x;
            this.y = y;
      }
```

```
	public Vektor() {
		this(0, 0);
	}

	public double dlzka() {
		return Math.sqrt(x * x + y * y);
	}

	public static Vektor sucet(Vektor v1, Vektor v2) {
		if ((v1 == null) || (v2 == null))
			return null;

		return new Vektor(v1.x + v2.x, v1.y + v2.y);
	}
}
```

Úlohy:

- V definícii triedy vyznačte konštruktory. Uveďte ich počet:
- Vyznačte v kóde triedy volania statických metód. Uveďte ich počet:
- V nasledujúcich častiach programu nájdite chybu (ak nejaká je) a vysvetlite ju.

Fragment programu	Vysvetlenie chyby (a možnosti odstránenia)
`Vektor v = new Vektor(3, 4);` `Int dlz = v.dlzka();`	
`Vektor a = Vektor(3, 4);` `a = Vektor.sucet(a, a);`	
`System.out.print(Vektor.dlzka())`	

- Čo vypíše nasledujúci fragment programu ? Vysvetlite.

```
Vektor u = new Vektor();
Vektor w = new Vektor(2, 3);
u.x = 4; w = u; w.y = 6;
System.out.println("u = [" + u.x + "," + u.y + "]");
```

Vytvorte triedu *MojVektor* oddedenú od triedy *Vektor* tak, aby nasledujúci fragment programu fungoval presne ako je uvedené:

```
// Vytvorime vektor v s hodnotou [2, 0]
MojVektor v = new MojVektor(2);
// Vynasobime vektor skalarom, vo v bude [6, 0]
v.vynasobSkalarom(3);
// Vytvorime vektor w s hodnotou [4, 5]
Vektor w = new MojVektor(4, 5);
// Vypocita skalarny sucit vektorov 6 * 4 + 5 * 0 = 24
double sucin = MojVektor.sucin(v, w);
// Vypise do konzoly retazec v = (4, 5)
System.out.println("v = " + v.toString());
```

Písomka 29. 11. 2007 B Meno:

I. V tejto otázke predpokladáme stručné a jasne formulované odpovede. Maximálne tri vety na každú otázku. Ak je potrebné, ilustrujte na príklade. Každá správna odpoveď je za 1 bod, okrem otázky č.3. Celá otázka je za 10 bodov.

1. Aké je trvanie platnosti lokálnej premennej?
2. Aký je rozsah hodnôt premennej primitívneho typu?
3. Vymenujte hlavné komponenty triedy a popíšte, čo je ich úlohou? - 2 body
4. Čo sa stane, keď metóda obsahuje lokálnu premennú, ktorá má rovnaké meno ako premenná inštancie v triede tejto metódy?
5. Aká metóda je rekurzívna?
6. Čo sú to preťažené metódy?
7. Aký je rozdiel v chovaní sa príkazov break a continue?
8. Aký je rozdiel medzi príkazmi `while` a `do-while`?
9. Čo je to referencia?

II. 10 bodov
Implementujte nasledovnú metódu:

Palindromizovaný reťazec

Metóda: `public String palindromizuj(String s)`

Opis: Vráti reťazec, ktorý vznikne prilepením reverzu reťazca *s* k pôvodnému reťazcu *s*.
`palindromizuj("Ahoj") -> "Ahojjohā"`

III. Uvažujme nasledujúcu triedu: (12 bodov)

```
public class Vektor {
	public double x;
	public double y;

	public Vektor(double x, double y) {
		this.x = x;
		this.y = y;
	}

	public Vektor() {
		this(0, 0);
	}

	public double dlzka() {
		return Math.sqrt(x * x + y * y);
	}

	public static Vektor sucet(Vektor v1, Vektor v2) {
		if ((v1 == null) || (v2 == null))
			return null;

		return new Vektor(v1.x + v2.x, v1.y + v2.y);
	}
}
```

Úlohy:

- V definícii triedy vyznačte konštruktory. Uveďte ich počet:

- Vyznačte v kóde triedy volania statických metód. Uveďte ich počet:
- V nasledujúcich častiach programu nájdite chybu (ak nejaká je) a vysvetlite ju.

Fragment programu	Vysvetlenie chyby (a možnosti odstránenia)
`Vektor v = new Vektor(3, 4);` `Int dlz = v.dlzka();`	
`Vektor a = Vektor(3, 4);` `a = Vektor.sucet(a, a);`	
`System.out.print(Vektor.dlzka())`	

- Čo vypíše nasledujúci fragment programu ? Vysvetlite.

```
Vektor u = new Vektor();
Vektor w = new Vektor(2, 3);
u.x = 4; w = u; w.y = 6;
System.out.println("u = [" + u.x + "," + u.y + "]");
```

Vytvorte triedu *MojVektor* oddedenú od triedy *Vektor* tak, aby nasledujúci fragment programu fungoval presne ako je uvedené:

```
// Vytvorime vektor v s hodnotou [2, 0]
MojVektor v = new MojVektor(2);
// Vynasobime vektor skalarom, vo v bude [6, 0]
v.vynasobSkalarom(3);
// Vytvorime vektor w s hodnotou [4, 5]
Vektor w = new MojVektor(4, 5);
// Vypocita skalarny sucit vektorov 6 * 4 + 5 * 0 = 24
double sucin = MojVektor.sucin(v, w);
// Vypise do konzoly retazec v = (4, 5)
System.out.println("v = " + v.toString());
```

IV. 8 bodov

Zistite, akú hodnotu vypočíta neznáma metóda pre 2 zadané vstupy a nájdite ľubovoľnú takú hodnotu parametra *n*, aby funkcia vypočítala zadaný výsledok (Vyplňte tabuľku).

```
public int vypocet(int n) {
      int v = 1;
      while (n > 0) {
            v = v * (n % 10);
            n = n / 10;
      }
      return v;
}
```

n	vypocet(n)
234	
1623	
	36

Písomka 29. 11. 2007 C Meno:

I. 8 bodov
Implementujte nasledovnú metódu:

Najvzdialenejšia dvojica

Metóda: `public double najvzdialenejsie(Turtle[] korytnacky)`

Opis: Vráti vzdialenosť medzi 2 najvzdialenejšími korytnačkami v poli *korytnacky*.
V poli *korytnacky* je každá korytnačka práve raz, žiaden prvok poľa nie je *null* a pole má dĺžku aspoň 5.

Niektoré užitočné metódy objektov triedy *Turtle*	
`double getX()`	Vráti x-ovú súradnicu aktuálnej pozície korytnačky.
`double getY()`	Vráti y-ovú súradnicu aktuálnej pozície korytnačky.
`double distanceTo(double x, double y)`	Vráti vzdialenosť korytnačky k bodu [x, y]

II. Uvažujme nasledujúcu triedu: (12 bodov)

```
public class Vektor {
	public double x;
	public double y;

	public Vektor(double x, double y) {
		this.x = x;
		this.y = y;
	}

	public Vektor() {
		this(0, 0);
	}

	public double dlzka() {
		return Math.sqrt(x * x + y * y);
	}

	public static Vektor sucet(Vektor v1, Vektor v2) {
		if ((v1 == null) || (v2 == null))
			return null;

		return new Vektor(v1.x + v2.x, v1.y + v2.y);
	}
}
```

Úlohy:

- V definícii triedy vyznačte konštruktory. Uveďte ich počet:
- Vyznačte v kóde triedy volania statických metód. Uveďte ich počet:
- V nasledujúcich častiach programu nájdite chybu (ak nejaká je) a vysvetlite ju.

Fragment programu	Vysvetlenie chyby (a možnosti odstránenia)
`Vektor v = new Vektor(3, 4);` `Int dlz = v.dlzka();`	
`Vektor a = Vektor(3, 4);` `a = Vektor.sucet(a, a);`	
`System.out.print(Vektor.dlzka())`	

- Čo vypíše nasledujúci fragment programu ? Vysvetlite.

```
Vektor u = new Vektor();
Vektor w = new Vektor(2, 3);
u.x = 4; w = u; w.y = 6;
System.out.println("u = [" + u.x + "," + u.y + "]");
```

Vytvorte triedu *MojVektor* oddedenú od triedy *Vektor* tak, aby nasledujúci fragment programu fungoval presne ako je uvedené:

```
// Vytvorime vektor v s hodnotou [2, 0]
MojVektor v = new MojVektor(2);
// Vynasobime vektor skalarom, vo v bude [6, 0]
v.vynasobSkalarom(3);
// Vytvorime vektor w s hodnotou [4, 5]
Vektor w = new MojVektor(4, 5);
// Vypocita skalarny sucit vektorov 6 * 4 + 5 * 0 = 24
double sucin = MojVektor.sucin(v, w);
// Vypise do konzoly retazec v = (4, 5)
System.out.println("v = " + v.toString());
```

III. V tejto otázke predpokladáme stručné a jasne formulované odpovede. Maximálne tri vety na každú otázku. Každá správna odpoveď je za 1 bod. Celá otázka je za 10 bodov.

1. Čo je pole? Z čoho sa pole skladá?
2. Čo je to referencia? Ako deklarujeme referenciu na pole?
3. Ako sa vytvára pole?
4. Ako je možné pole inicializovať?
5. Predpokladajte, že ste vytvorili pole, ktoré má 100 prvkov. Aký interval indexov je možné použiť, ak sa chceme odvolávať na prvky tohto poľa?
6. Určte, ktoré z nasledujúcich výrazov Javy sú spravné. U každého výrazu určte, čo sa bude v programe robiť:

e)
```
double arr[]; arr=new double[10];
```
f)
```
double aaa[]; aaa=new int[100];
```
g)
```
double bbb[]; bbb[]={1., 2., 3., 4., 5., 6.};
```
h)
```
double aaa[] = {1., 2., 3., 4., 5.};
for (i=1; i<aaa.length; i++)
   System.out.println("i = " + aaa[i]);
```
e)
```
double aaa[] = {1., 2., 3., 4., 5.};
for (i=aaa.length-1; i>=0; i--)
   System.out.println("i = " + aaa[i]);
```

IV. 10 bodov
Implementujte nasledovnú metódu:

Premixovaný reťazec

Metóda:	`public String premixuj(String s)`
Opis:	Vráti reťazec, ktorý vznikne zlepením 2 reťazcov: prvý reťazec pozostáva z písmen na párnych pozíciách reťazca *s* a druhý reťazec z písmen na nepárnych pozíciách. `premixuj("Programovanie") -> "Pormvnergaoai"`

Písomka 29. 11. 2007 D Meno:

I. Uvažujme nasledujúcu triedu: (12 bodov)

```
public class Vektor {
	public double x;
	public double y;

	public Vektor(double x, double y) {
		this.x = x;
		this.y = y;
	}

	public Vektor() {
		this(0, 0);
	}

	public double dlzka() {
		return Math.sqrt(x * x + y * y);
	}

	public static Vektor sucet(Vektor v1, Vektor v2) {
		if ((v1 == null) || (v2 == null))
			return null;

		return new Vektor(v1.x + v2.x, v1.y + v2.y);
	}
}
```

Úlohy:

- V definícii triedy vyznačte konštruktory. Uveďte ich počet:
- Vyznačte v kóde triedy volania statických metód. Uveďte ich počet:
- V nasledujúcich častiach programu nájdite chybu (ak nejaká je) a vysvetlite ju.

Fragment programu	Vysvetlenie chyby (a možnosti odstránenia)
`Vektor v = new Vektor(3, 4);` `Int dlz = v.dlzka();`	
`Vektor a = Vektor(3, 4);` `a = Vektor.sucet(a, a);`	
`System.out.print(Vektor.dlzka())`	

- Čo vypíše nasledujúci fragment programu ? Vysvetlite.

```
Vektor u = new Vektor();
Vektor w = new Vektor(2, 3);
u.x = 4; w = u; w.y = 6;
System.out.println("u = [" + u.x + "," + u.y + "]");
```

Vytvorte triedu *MojVektor* oddedenú od triedy *Vektor* tak, aby nasledujúci fragment programu fungoval presne ako je uvedené:

```
// Vytvorime vektor v s hodnotou [2, 0]
MojVektor v = new MojVektor(2);
// Vynasobime vektor skalarom, vo v bude [6, 0]
v.vynasobSkalarom(3);
```

```
        // Vytvorime vektor w s hodnotou [4, 5]
        Vektor w = new MojVektor(4, 5);
        // Vypocita skalarny sucit vektorov 6 * 4 + 5 * 0 = 24
        double sucin = MojVektor.sucin(v, w);
        // Vypise do konzoly retazec v = (4, 5)
        System.out.println("v = " + v.toString());
```

II. 8 bodov

Zistite, akú hodnotu vypočíta neznáma metóda pre 2 zadané vstupy a nájdite ľubovoľnú takú hodnotu parametra *n*, aby funkcia vypočítala zadaný výsledok (Vyplňte tabuľku).

```
public int vypocet(int n) {
        int v = 1;
        while (n > 0) {
                v = v * (n % 10);
                n = n / 10;
        }
        return v;
}
```

n	vypocet(n)
234	
1623	
	36

III. V tejto otázke predpokladáme stručné a jasne formulované odpovede. Maximálne tri vety na každú otázku. Ak je potrebné, ilustrujte na príklade. Každá správna odpoveď je za 1 bod, okrem otázky č.3. Celá otázka je za 10 bodov.

1. Aké je trvanie platnosti lokálnej premennej?
2. Aký je rozsah hodnôt premennej primitívneho typu?
3. Vymenujte hlavné komponenty triedy a popíšte, čo je ich úlohou? - 2 body
4. Čo sa stane, keď metóda obsahuje lokálnu premennú, ktorá má rovnaké meno ako premenná inštancie v triede tejto metódy?
5. Aká metóda je rekurzívna?
6. Čo sú to preťažené metódy?
7. Aký je rozdiel v chovaní sa príkazov break a continue?
8. Aký je rozdiel medzi príkazmi `while` a `do-while`?
9. Čo je to referencia?

IV. 10 bodov

Implementujte nasledovnú metódu:

Palindromizovaný reťazec

Metóda: `public String palindromizuj(String s)`

Opis: Vráti reťazec, ktorý vznikne prilepením reverzu reťazca *s* k pôvodnému reťazcu *s*.
`palindromizuj("Ahoj") -> "Ahojjohʌ"`