

Laboratorium 6



CEL: Hermetyzacja. Dziedziczenie. Polimorfizm.



Zadanie#1. Przeanalizuj poniższy kod.

```
public class Example {  
    public Example() { }  
  
    public void cutMaterial( ) {  
        System.out.println( "Cut Material" );  
    }  
}
```

Przekształć kod klasy do postaci reprezentującej wzorec projektowy Singleton. Przedstaw kod całościowo.



Zadanie#2. Które z poniższych zdań jest/są prawdziwe?

a)	Wszystkie klasy Java dziedziczą z klasy Object
b)	Jeśli tego jawnie nie wskazano klasa Java nie dziedziczy z żadnej innej klasy
c)	Wszystkie klasy Java dziedziczą z klasy Runnable
d)	W Java nie ma dziedziczenia wielobazowego



Zadanie#3. Czy poniższa deklaracja klasy Test3 jest poprawna (z założeniem, że klasa Test2 i interfejs Test1 zostały już zdefiniowane)?

```
public class Test3 extends Test2 implements Test1 {}
```

a)	Tak
b)	Nie



Zadanie#4. Przeanalizuj poniższy kod.

```
public class EncapsulationTest {  
    int encapOne;  
    int encapTwo;  
    int encapThree;  
  
    //encapsulation code//  
}
```

Ze względu na zasadę hermetyzacji należałoby (jedna odpowiedź):

a)	Wszystkie pola oznaczyć jako <code>private</code>
b)	Wszystkie pola oznaczyć jako <code>public</code>
c)	Wszystkie pola oznaczyć jako <code>protected</code>
d)	Wszystko jest poprawnie





Zadanie#5. Przeanalizuj poniższy kod.

```
class Base {
    void methodA() {
        System.out.println("base - MethodA");
    }
}

class Sub extends Base {
    public void methodA() {
        System.out.println("sub - MethodA");
    }

    public void methodB() {
        System.out.println("sub - MethodB");
    }

    public static void main(String args[]) {
        Base b = new Sub(); // Line 1
        b.methodA(); // Line 2
        b.methodB(); // Line 3
    }
}
```

Jaki będzie wynik wykonania?

a)	sub - MethodA , sub - MethodB
b)	base - MethodA , sub - MethodB
c)	Błąd kompilacji w // Line 1
d)	Błąd kompilacji w // Line 2
e)	Błąd kompilacji w // Line 3



Zadanie#6. Przeanalizuj poniższy kod.

```
01: public class OverTest {
02:     public int add(int a, int b) {
03:         return a + b;
04:     }
05:
06:     public static long add2(long a, long b) {
07:         return a + b;
08:     }
09: }
10:
11: public class OverTest2 extends OverTest {
12:     public int add(int a, int b) {
13:         return a - b;
14:     }
15:
16:     public static long add2(long a, long b) {
17:         return a - b;
18:     }
19: }
```

Jaki mechanizm został wykorzystany dla metod `add` oraz `add2` (nadklasa/podklasa)? Uzupełnij kod zamieszczając stosowne adnotacje Java (przedstaw kod całościowo).

a)	Overloading dla <code>add</code>
b)	Overriding dla <code>add</code>
c)	Overloading dla <code>add2</code>
d)	Overriding dla <code>add2</code>
e)	Overloading dla <code>add</code> i <code>add2</code>
f)	Overriding dla <code>add</code> i <code>add2</code>
g)	Overloading dla <code>add</code> i overriding <code>add2</code>
h)	Overriding dla <code>add</code> i overloading <code>add2</code>



Zadanie#7. Przeanalizuj poniższy kod.

```

01: class Man {
02:     String name;
03:     int    id;
04:
05:     public Man( ) {
06:         System.out.println("1");
07:     }
08: }
09:
10: class Employee extends Man {
11:     String title;
12:     {
13:         System.out.println("4");
14:     }
15:     static {
16:         System.out.println("2");
17:     }
18:
19:     public Employee() {
20:         System.out.println("3");
21:     }
22: }
23:
24: public class TestOrder {
25:     public static void main( String[ ] args ) {
26:         Employee obj = new Employee( );
27:     }
28: }

```

Wskaż, jaki wynik zostanie wyświetlony na konsoli?

a)	1 2 3 4
----	------------------

b)	4 2 3 1
c)	2 1 4 3
d)	1 2 3



Zadanie#8. Przeanalizuj poniższy kod.

```
package test;

class Test {
    [modifier] int i;
    /* lot of code */
}
```

Jakiego modyfikatora dostępu należałoby użyć dla pola `i`, aby nie było ono widoczne poza pakietem `test`, ale było widoczne wewnątrz tego pakietu?

a)	friend
b)	public
c)	protected
d)	Nic nie wpisywać



Zadanie#9. Czy definiując tablicę typem obiektów przechowywanych w jej wnętrzu może być klasa abstrakcyjna lub interfejs?

a)	Tak
b)	Nie



Zadanie#10. Czy klasa abstrakcyjna może mieć konstruktor?

a)	Tak
b)	Nie



Zadanie#11. Przeanalizuj poniższy kod.

```
01: if (null instanceof Object) {
02:     System.out.println("1");
03: }
04: else {
05:     System.out.println("2");
06: }
```

Jaki będzie wynik wykonania?

a)	1
b)	2
c)	NullPointerException
d)	Compilation error (Incompatible conditional operand types)



Zadanie#12. Czy metoda abstrakcyjna może być prywatna lub finalna?

a)	Tak
b)	Nie



Zadanie#13. Przeanalizuj poniższy kod.

```
import java.util.Random;

interface Figura {
    public void rysuj( );
}

class Wielokat implements Figura {
    public void rysuj( ) {
        System.out.print( "Wielokat.rysuj( ) - " );
    }
}

class Elipsa implements Figura {
    public void rysuj( ) {
        System.out.print( "Elipsa.rysuj( ) - " );
    }
}

public class PolymorphismExample {
    public static void main( String[] args ) {
        Random random = new Random( );
        Figura[ ] figura = new Figura[ 9 ];

        for( int i = 0; i < figura.length; i++ ) {
            figura[i] = ( random.nextInt( 2 )==0 ) ? new Wielokat( )
                                                    : new Elipsa( );
        }

        //TODO: Rozpoznawanie typu obiektu w tablicy figura
    }
}
```

Dopisz w miejscu //TODO kod rozpoznający za pomocą operatora instanceof typ obiektu w kolejnych elementach tablicy figura. Przedstaw całościowo ostateczny kod.



Zadanie#14. Przeanalizuj poniższy kod.

```
public abstract class Example {
    public abstract void transform();
}
```

Na podstawie klasy `Example` zdefiniuj dwie dowolnie nazwane podklasy i zdefiniuj w każdej z nich (dowolnie) metodę polimorficzną `transform()`. Przedstaw całościowo ostateczny kod.



Zadanie#15. Przeanalizuj poniższy kod.

```
01: interface IA { int x = 1; }
02: class A { int x = 2; }
03:
04: class B extends A implements IA {
05:
06:     public static void main(String args[]) {
07:         System.out.println(x);
08:     }
09: }
```

Jaki będzie wynik wykonania?

a)	0
b)	1
c)	2
d)	Błąd kompilacji



Poprawnych odpowiedzi	Ocena
<0>	n/k
<1, 8>	2.0
<9, 10>	3.0
<11>	3.5
<12, 13>	4.0
<14>	4.5
<15>	5.0