תוכנה 1

תרגול 13 - שונות

הדס צור ואסף זריצקי

קצת על מנשקים

- מנשק יכול להרחיב **יותר ממנשק אחד**
- שירותים במנשק הם תמיד **מופשטים וציבוריים**

```
public interface MyInterface {
   public abstract int foo1(int i);
   int foo2(int i);
}
```


מנשקים

```
public interface Foo {
    public void bar() throws Exception;
}

public class FooImpl implements Foo {
    public void bar() {
        System.out.println("No  exception is thrown");
    }

    public static void main(String args[]) {
        Foo foo = new FooImpl();
        foo.bar();
    }
}
```

Compilation Error:
"Unhandled exception type Exception"
If yes, why? If no, what is the output?

מנשקים

```
public interface Foo {
    public void bar() throws Exception;
}

public class FooImpl implements Foo {
    public void bar() {
        System.out.println("No exception is thrown");
    }

    public static void main(String args[]) {
        FooImpl foo = new FooImpl();
        foo.bar();
    }
}
```

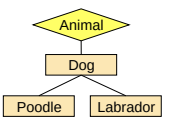
Output:
No exception is thrown
If yes, why? If no, what is the output?

http://www.sourcebank.com/tips/Tip/14463

מנשקים וירושה

Consider the following class hierarchy:

```
Interface Animal {…}
class Dog implements Animal{…}
class Poodle extends Dog {…}
class Labrador  extends Dog {…}
```



Which of the following lines (if any) will not compile?

```
Poodle poodle = new Poodle();
Animal animal = (Animal) poodle;
Dog dog = new Labrador();
animal = dog;
poodle = dog;
```

poodle = (Poodle) dog;
-No compilation error
-Runtime Exception

- Compilation Error
Type mismatch: cannot convert

Labrador labrador = (Labrador) animal;
-No compilation error
-No Runtime Exception

מנשקים וירושה

```
class A {
   public void print() {
      System.out.println("A");
   }
}

class B extends A implements C {
}

interface C {
   void print();
}
```

Is there an error?

public by default

מנשקים וירושה

```
class A {
   void print() {
      System.out.println("A");
   }
}

class B extends A implements C {
}

interface C {
   void print();
}
```

Is there an error?
The inherited package method A.print() cannot hide the public abstract method in C

Method Overloading & Overriding

```
public class A {
    public float foo(float a, float b) throws IOException{
    }
}

public class B extends A {
    …
}
```

Which of the following methods can be defined in B:

1. float foo(float a, float b){…}
2. public int foo(int a, int b) throws Exception{…}
3. public float foo(float a, float b) throws Exception{…}
4. public float foo(float p, float q) {…}

Answer: 2 and 4

Method Overriding

```
public class A {
   public void print() {
      System.out.println("A");
   }
}

public class B extends A {
   public void print(){
      System.out.println("B");
   }
}
```

```
public class C {
    public static void main(String args[]){
        B b = new B();
        A a = b;

        b.print();
        a.print();
    }
}
```

Casting is unneeded

The output is:
B
B

compile? If no, why?
throw a runtime exception?
o, what is the output?

Method Overriding & Visibility

```
public class A {
    public void print() {
        System.out.println("A");
    }
}

public class B extends A {
    protected void print() {
        System.out.println("B");
    }
}
```

```
public class C {
    public static void main(String[] args) {
        B b = new B();
        b.print();
    }
}
```

Compilation error:
"Cannot reduce the visibility of the inherited method"

no, why?
time exception?
he output?

Method Overriding & Visibility

```
public class A {
    protected void print() {
        System.out.println("A");
    }
}

public class B extends A {
    public void print() {
        System.out.println("B");
    }
}
```

```
public class C {
    public static void main(String[] args) {
        B b = new B();
        b.print();
    }
}
```

The output is:
B

?

Inheritance

```
public class A {
    public void foo() {
        System.out.println("A.foo()");
    }

    public void bar() {
        System.out.println("A.bar()");
        foo();
    }
}
```

```
public class B extends A {
    public void foo() {
        System.out.println("B.foo()");
    }

    public static void main(String[] args) {
        A a = new B();
        a.bar();
    }
}
```

The output is:
A.bar()
B.foo()

lle? If no, why?
a runtime exception?
at is the output?

Inheritance

```
public class A {
    private void foo() {
        System.out.println("A.foo()");
    }

    public void bar() {
        System.out.println("A.bar()");
        foo();
    }
}
```

```
public class B extends A {
    public void foo() {
        System.out.println("B.foo()");
    }

    public static void main(String[] args) {
        A a = new B();
        a.bar();
    }
}
```

Does the code compile? If no, why?
Does the code throw a runtime exception?
If yes, why? If no, what is the output?

Inheritance

```
public class A {
    public void foo() {…}
}

public class B extends A {
    public void foo() {…}
}
```

How can you invoke the foo method of A within B?
Answer:
Use super.foo()

Inheritance

```
public class A {
    public void foo() {…}
}

public class B extends A {
    public void foo() {…}
}

public class C extends B {
    public void foo() {…}
}
```

How can you invoke the foo method of A within C?
Answer:
Not possible
(super.super.foo() is illegal)

Inheritance & Constructors

```
public class A {
    String bar = "A.bar";

    A() { foo(); }

    public void foo() {
        System.out.println("A.foo(): bar = " + bar);
    }
}

public class B extends A {
    String bar = "B.bar";

    B() { foo(); }

    public void foo() {
        System.out.println("B.foo(): bar = " + bar);
    }
}
```

```
public class D {
    public static void main(String[] args) {
        A a = new B();
        System.out.println("a.bar = " + a.bar);
        a.foo();
    }
}
```

The output is:
B.foo(): bar = null
B.foo(): bar = B.bar
a.bar = A.bar
B.foo(): bar = B.bar

Inheritance & Constructors

```
public class A {
    protected B b = new B();
    public A() { System.out.println("in A: no args."); }
    public A(String s) { System.out.println("in A: s = " + s); }
}

public class B {
    public B() { System.out.println("in B: no args."); }
}

public class C extends A {
    protected B b;
    public C() { System.out.println("in C: no args."); }
    public C(String s) { System.out.println("in C: s = " + s); }
}

public class D {
    public static void main(String args[]) {
        C c = new C();
        A a = new C();
    }
}
```

The output is:
in B: no args.
in A: no args.
in C: no args.
in B: no args.
in A: no args.
in C: no args.

Inheritance & Constructors

```
public class A {
    protected B b = new B();
    public A() { System.out.println("in A: no args."); }
    public A(String s) { System.out.println("in A: s = " + s); }
}

public class B {
    public B() { System.out.println("in B: no args."); }
}

public class C extends A {
    protected B b;
    public C() { System.out.println("in C: no args."); }
    public C(String s) { System.out.println("in C: s = " + s); }
}

public class D {
    public static void main(String args[]) {
        C c = new C("c");
        A a = new C("a");
    }
}
```

The output is:
in B: no args.
in A: no args.
in C: s = c
in B: no args.
in A: no args.
in C: s = a

Inheritance & Constructors

```
public class A {
    protected B b = new B();
    public A() { System.out.println("in A: no args."); }
    public A(String s) { System.out.println("in A: s = " + s); }
}

public class B {
    public B() { System.out.println("in B: no args."); }
}

public class C extends A {
    protected B b;
    public C() { System.out.println("in C: no args."); }
    public C(String s) { System.out.println("in C: s = " + s); }
}

public class D {
    public static void main(String args[]) {
        C c = new C("c");
        A a = new C("a");
    }
}
```

What will happen if we remove this line?

Inheritance & Constructors

```
public class A {
    String bar = "A.bar";
}

public class B extends A {
    String bar = "B.bar";

    B() { foo(); }

    public void foo() {
        System.out.println("B.foo(): bar = " + bar);
    }

    public static void main(String[] args) {
        A a = new B();
        System.out.println(a.bar);
        a.foo();
    }
}
```

Will this compile?
Will there be a RTE?
What is the result?

בחינה באופק!

- הבחינה ב- 26.6
- כל הנושאים שכיסינו במהלך הסמסטר (שיעורים, תרגולים ועבודות בית)
  - Java, DBC, ירושה ופולימורפיזם, iterator, IO, Generics, מחלקות פנימיות, Collection Framework, ...
- לפתור כמה שיותר מבחנים משנים שעברו
  - לא כל הסמסטרים זהים מבחינת החומר

בהצלחה!