1. Operacje wejścia/wyjścia na plikach można realizować w Javie za pomocą dwóch API: IO oraz NIO. Jak można pozyskać listę wszystkich plików (bez katalogów) znajdujących się na dysku?

a) metodą klasy `Files` (NIO)

```
public static Stream<Path> list(Path dir) throws IOException
```

b) metodą klasy `File` (IO)

```
public String[] list()
```

c) metodą klasy `File` (IO)

```
public File[] listFiles()
```

d) klasy `Files` (NIO) oraz `File` (IO) nie dostarczają metod do pozyskiwania listy samych tylko plików (zawsze trzeba je odfiltrowywać)

2. Czy wykonanie poniższego kodu spowoduje automatyczne utworzenie nowego pliku na dysku (przy założeniu, że wszystkie importy zostały prawidłowo zadeklarowane)?

```
public class A {
    public static void main(String[] args) {
      File f = new File("in.txt");
    }
```

a) Nie – `File` to klasa reprezentująca ścieżki dostępu do plików bądź katalogów (te pliki i katalogi wcale nie muszą istnieć na dysku).
b) Nie – jeśli plik o podanej nazwie istnieje, to nowy nie zostanie utworzony, jeśli zaś takiego pliku nie ma, to powstanie nowy.
c) Nie – obiekt typu `File` służy do przechowywania danych w pamięci, które później można zapisać na dysku.
d) Tak – utworzenie obiektu typu `File` zawsze skutkuje utworzeniem nowego pliku.

3. Która z metod klasy `File` pozwala usunąć plik z dysku?

a) `delete()`
b) `release()`
c) `remove()`
d) klasa `File` nie posiada metod pozwalających usunąć plik

4. Co można powiedzieć o poniższym kodzie (dla jdk 1.8 i wyżej)?

```
public class A <T> {
    public void h(T i) {          // 1
        System.out.println(i);
    }

    public static void main(String[] args) {
      A<Integer> ai = new A<>(); // 2
      A<> as = new A<String>();  // 3
    }
}
```

a) Jest to niepoprawny kod (zawiera błąd w linii 1)
b) Jest to niepoprawny kod (zawiera błąd w linii 2)
c) Jest to niepoprawny kod (zawiera błąd w linii 3)
d) Jest to poprawny kod

5. Co można powiedzieć o poniższym kodzie?

```
public class B {
    interface I {
        void m();
        void n();
    }
    public I metoda;

    public static void main(String[] args) {
        B b = new B();
        b.metoda = ()-> System.out.println("OK");
    }
}
```

a) Jest to niepoprawny kod (nie można deklarować interfejsów wewnątrz klas)
b) Jest to niepoprawny kod (błąd występuje w linii z przypisaniem wyrażenie lambda)
c) Jest to poprawny kod, jednak jego wykonanie zakończy się zgłoszeniem wyjątku
d) Jest to poprawny kod, a po jego wykonaniu ekran pozostanie pusty

6. Co można powiedzieć o poniższym kodzie?

```
interface I<T> {
    void n();
}

public class A {
    public static I metoda = ()-> {};// 1

    public static void main(String[] args) {
        metoda();                    // 2
    }
}
```

a) Jest to niepoprawny kod (źle zadeklarowano interfejs)
b) Jest to niepoprawny kod (błąd występuje w linii 1)
c) Jest to niepoprawny kod (błąd występuje w linii 2)
d) Jest to poprawny kod

7. Co można powiedzieć o poniższym kodzie?

```
interface I {
    static void n(int i) {System.out.print(i);};
}

public class A {
    public static void main(String[] args) {
        IntStream.iterate(0, i -> i + 2).
           limit(3).
            forEach(I::n);
    }
}
```

a) Jest to niepoprawny kod
b) Jest to poprawny kod, po jego uruchomieniu na ekranie zostanie wypisane: `02`
c) Jest to poprawny kod, po jego uruchomieniu na ekranie zostanie wypisane: `246`
d) Jest to poprawny kod, po jego uruchomieniu na ekranie zostanie wypisane: `024`

8. Co można powiedzieć o poniższym kodzie?

```
interface I {
    default void m() {}
}
public class A {
    public class B {
        I i;
    }
    public static void main(String[] args) {
        Thread t1 = new Thread ((new A()).new B().i::m);
        Thread t2 = new Thread (A.B.i::m);
        Thread t3 = new Thread (I::m);
    }
}
```

a) Jest to niepoprawny kod (źle zadeklarowano wątek `t1`)
b) Jest to niepoprawny kod (źle zadeklarowano wątek `t2`)
c) Jest to niepoprawny kod (źle zadeklarowano wątek `t3`)
d) Jest to poprawny kod

9. Co można powiedzieć o poniższym kodzie?

```
abstract class B {
}

public class A extends B {
    public static void main(String[] args) {
    }
}
```

a) Jest to poprawny kod
b) Jest to niepoprawny kod (nie można przy dziedziczeniu rozszerzać dostępu)
c) Jest to niepoprawny kod (nie można deklarować klasy abstrakcyjnej bez żadnej metody abstrakcyjnej)
d) Jest to niepoprawny kod (klasa `A` powinna być zadeklarowana jako klasa abstrakcyjna)

10. Co można powiedzieć o poniższym kodzie?

```
import java.util.Arrays;
import java.util.TreeSet;

public class A {
    public static void main(String[] args) {
        TreeSet<String> ts = new TreeSet<String>();
        ts.addAll(Arrays.asList("B","C","D","A","D","C"));
        System.out.println(ts);
    }
}
```

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `[A, B, C, D]`
b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `[B, C, D, A, D, C]`
c) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `[D, C, B, A]`
d) Jest to niepoprawny kod

11. Co można powiedzieć o poniższym kodzie?

```
import java.util.Collections;
import java.util.TreeSet;

public class A {
    public static void main(String[] args) {
        TreeSet<Integer> ts = new TreeSet<Integer>();
        Collections.addAll(ts, new int[] {3,1,2,3,1});
        System.out.println(ts);
    }
}
```

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `[1, 2, 3]`
b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `[3, 1, 2, 3, 1]`
c) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `[3, 2, 1]`
d) Jest to niepoprawny kod

12. Co można powiedzieć o poniższym kodzie?

```
public class A <T> {
     T t = new T();                    // 1

    public static void main(String[] args) {
        A<String> a = new A<>();  // 2
    }
}
```

a) Jest to poprawny kod, wykonanie metody `main` przebiegnie poprawnie
b) Jest to poprawny kod, jednak podczas uruchomienia metody `main` zostanie wyrzucony wyjątek
c) Jest to niepoprawny kod (błąd występuje w linii `1`)
d) Jest to niepoprawny kod (błąd występuje w linii `2`)

13. Co można powiedzieć o poniższym kodzie?

```
class B extends A { public String c = "B"; }
class D extends B { public String c = "D"; }

public class A {
    public String c = "A";

   public static void main(String[] args) {
        System.out.println(new A().c + new B().c + new D().c);
    }
}
```

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `ABD`

b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `AAA`

c) Jest to niepoprawny kod (nie można deklarować w klasach potomnych pól o tej samej nazwie co pole w klasie bazowej)

d) Jest to niepoprawny kod (nie można deklarować klas potomnych dopóki nie zostanie zdefiniowana klasa bazowa)

14. Co można powiedzieć o poniższym kodzie?

```
public class A {
    public String c = "A";
    public String getC() { return c;}

    public static void main(String[] args) {
        new B().<B>m(new B(), new D());
    }
}

class B extends A {
    public String c = "B";
    <V extends B> void m(V a1, V a2) {
        System.out.println(a1.getC()+a2.getC());
    }
}

class D extends B { public String c = "D"; }
```

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `BD`
b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `BB`
c) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `AA`
d) Jest to niepoprawny kod

15. Co można powiedzieć o poniższym kodzie?

```
class E1 extends Exception {}
class E2 extends E1 {}
interface I {
    void m1() throws E1;
    void m2() throws E2;
}

public class A implements I {
    @Override
    public void m1() throws E2{} // 1
    @Override
    public void m2() throws E1{} // 2
}
```

a) Jest to poprawny kod.
b) Jest to niepoprawny kod (źle zadeklarowany klasy wyjątków `E1` i `E2`)
c) Jest to niepoprawny kod (źle zaimplementowano metodę `m1` w linii `1`)
d) Jest to niepoprawny kod (źle zaimplementowano metodę `m2` w linii `2`)

16. Co można powiedzieć o poniższym kodzie?

```
public class A {
    public static int m(int i) throws Exception{
        if (i<=0) new Exception("Exception ");
        System.out.print("No exception ");
        return i++;
    }
    public static void main(String[] args) {
        int i = 10;
        try {
            i=m(0);
        } catch (Exception e) {
            System.out.print(e.getMessage());
        } finally {
            System.out.print("Finally ");
        }
        System.out.println(i);
    }
}
```

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `No exception Finally 0`
b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `No exception Finally 1`
c) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `Exception Finally 10`
d) Jest to niepoprawny kod

17. Co można powiedzieć o poniższym kodzie?

`package p1;`
`class A {`
`public static void main(String ... args) {`
`    }`
`}`
	`package p1.p2;`
	`public class A {`
	`public static void main(String[] args) {`
	`        p1.A.main(args);`
	`    }`
	`}`

a) Jest to niepoprawny kod (klasa `p1.A` jest niewidoczna w pakiecie `p1.p2`)
b) Jest to niepoprawny kod (nie można wywołać metody `main` wewnątrz innej metody `main`)
c) Jest to niepoprawny kod (źle zadeklarowano metodę `main` w klasie `p1.A`)
d) Jest to poprawny kod. Można uruchomić program wskazując dowolną z tych dwóch klas jako klasę główną.

18. Co można powiedzieć o poniższym kodzie?

`package p1;`
`public class B {`
`}`
	`package p1.p2;`
	`import p1.*;`
	`public class A {`
	`  static public class B { }`
	`  public static void main(String[] args) {`
	`  System.out.println(new B());`
	`  }`
	`}`

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `p1.B@15dbaa42`
b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `p1.p2.A$B@15db9742`
c) Jest to niepoprawny kod (nie można importować wszystkich klas ze wskazanego pakietu, jeśli wśród nich jest klasa o nazwie pokrywającej się z nazwą klasy występującej w bieżącym kodzie)
d) Jest to niepoprawny kod (nie można do `println` wstawić obiektu utworzonego operatorem `new`)

19. Co można powiedzieć o poniższym kodzie?

```
public class A {
    private  int i ;
    public void m() { i = 10;}

    public static void main(String[] args) {
        try {
            A a = new A();
        } catch(Exception e) {}
        finally {
            a.m();
        }
    }
}
```

a) Jest to poprawny kod
b) Jest to niepoprawny kod (brak dostępu do zmiennej w bloku `finally`)
c) Jest to niepoprawny kod (nie można zakładać bloku try-catch na fragmencie kodu, w którym nie wyrzuca się żadnych wyjątków)
d) Jest to niepoprawny kod (nie można deklarować pola `i` jak wyżej bez jego zainicjalizowania)

20. Co można powiedzieć o poniższym kodzie?

```
public class A {
    A () { System.out.print("A");}
    A (int i) {}
    public static void main(String[] args) {
      A a = new B();
    }
}
class B extends A {
    public B() {
        System.out.print("B");
    }
}
```

a) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `A`
b) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `B`
c) Jest to poprawny kod. Po wykonaniu metody `main` na ekranie pojawi się: `AB`
d) Jest to niepoprawny kod

21. Co można powiedzieć o poniższym kodzie (zakładając, że wszystkie importy zostały poprawnie zadeklarowane)?

```java
public class A implements I, Serializable  {
    protected A() throws RemoteException {}
    @Override
    public void m(int i) {
        System.out.println(i);
    }

    public static void main(String[] a)
            throws RemoteException,
                   AlreadyBoundException {

        LocateRegistry.
          createRegistry(2000).
            bind("A", new A());
        while(true);
    }
}
```

```java
public interface I extends Remote{
   public void m(int i)
     throws RemoteException;
}
```

```java
public class B {
public static void main(String[] a)
            throws RemoteException,
                   NotBoundException {

  ((I) LocateRegistry.
         getRegistry(2000).
          lookup("A")).m(10);
            }
}
```

a) Jest to poprawny kod. Po uruchomieniu w różnych konsolach, kolejno, klasy `A` i klasy `B`, na konsoli klasy `B` pojawi się `10`

b) Jest to poprawny kod. Po uruchomieniu w różnych konsolach, kolejno, klasy `A` i klasy `B`, na konsoli klasy `A` pojawi się `10`

c) Jest to poprawny kod. Jednak po uruchomieniu w różnych konsolach, kolejno, klasy `A` i klasy `B`, na konsoli klasy `B` zostanie zgłoszony wyjątek

d) Jest to niepoprawny kod

22. Co można powiedzieć o poniższym kodzie?

```java
public class A {
public static String msg = "A";
public static void main(String[] args)
        throws IOException {
 ServerSocket s = new ServerSocket(2000);
 while (true) {
  Socket sc = s.accept();
  new PrintWriter(sc.getOutputStream()).
    println(msg);
  System.out.print(" S:" + msg);
  BufferedReader br =
    new BufferedReader(
        new InputStreamReader(
            sc.getInputStream()));
  System.out.print(" R:" + br.readLine());
  sc.close();
 }
}}
```

```java
public class B {
public static String msg = "B";
 public static void main(String[] a)
            throws IOException {
 Socket s;
 s = new Socket("localhost",2000);
 OutputStream out =
        s.getOutputStream();
 PrintWriter pw =
        new PrintWriter(out, false);
 pw.println(msg);
 pw.flush();
 pw.close();
 s.close();
}
}
```

a) Jest to poprawny kod. Po uruchomieniu w różnych konsolach, kolejno, klasy `A` i klasy `B`, na konsoli klasy `A` pojawi się `S:M`

b) Jest to poprawny kod. Po uruchomieniu w różnych konsolach, kolejno, klasy `A` i klasy `B`, na konsoli klasy `A` pojawi się `S:M R: B`

c) Jest to poprawny kod. Jednak po uruchomieniu w różnych konsolach, kolejno, klasy `A` i klasy `B`, na konsoli klasy `A` zostanie zgłoszony wyjątek

d) Jest to niepoprawny kod (źle utworzono gniazdo serwerowe)

23. Które ze zdań są prawdziwe?

a) Zawartość plików znajdujących się w archiwum jar może być bezpośrednio odczytywana (przez klasy, które też są w tym archiwum).
b) Zawartość plików znajdujących się w archiwum jar może być bezpośrednio nadpisywana (przez klasy, które też są w tym archiwum).
c) Jeśli pliki w archiwum jar mają być bezpośrednio odczytywane (przez klasy, które też są w tym archiwum), to muszą znajdować się w katalogu głównym tego archiwum.
d) Jeśli pliki w archiwum jar mają być bezpośrednio nadpisane (przez klasy, które też są w tym archiwum), to muszą znajdować się w katalogu głównym tego archiwum.

24. Co można powiedzieć o poniższym kodzie?

```
interface I { void m(); }
interface J { int m() throws Exception; }
public class A {
    void ma(I i) {
        i.m();
    }
    void ma(J i) {
        try {
            i.m();
        } catch (Exception e) {
        }
    }
    public static void main(String[] args) {
        A a = new A();
        a.ma(() -> {
            while (true)
                Thread.currentThread().sleep(100);
        });
    }
}
```

a) Jest to niepoprawny kod (nie można tak przeciążać metody `ma`)
b) Jest to niepoprawny kod (brak obsługi wyjątku przy wywołaniu metody `sleep`)
c) Jest to niepoprawny kod (nie można zidentyfikować, jaki interfejs reprezentuje zadeklarowane wyrażenie lambda przekazane do metody `ma`)
d) Jest to poprawny kod

25. Co można powiedzieć o poniższym kodzie?

```
import java.util.Arrays;
import java.util.stream.IntStream;

public class A {

    public static void main(String[] args) {
        int tab[] = new int[] {1, 0};
        IntStream.range(4, 6).
          forEach(x -> { tab[x % 2] = x + 2 * x; });
        System.out.println(Arrays.toString(tab));
    }
}
```

a) Jest to poprawny kod. Po uruchomieniu metody `main` na ekranie pojawi się: `[12, 15]`
b) Jest to poprawny kod. Po uruchomieniu metody `main` na ekranie pojawi się: `[13, 16]`
c) Jest to poprawny kod. Po uruchomieniu metody `main` na ekranie pojawi się: `[13, 15]`
d) Jest to niepoprawny kod

26. Co można powiedzieć o poniższym kodzie?

```
interface I {
    void run0();
    static void run1() {}
    default void run2() {}
    default void run3() {}
}
public class A {
    public static class B {
         static I i = ()->{};
    }
    public static void main(String[] args) {
        A.B.i.run0();
        A.B.i.run1();
        A.B.i.run2();
        A.B.i.run3();
    }
}
```

a) Jest to niepoprawny kod (źle zadeklarowano interfejs `I`)
b) Jest to niepoprawny kod (źle zadeklarowano klasę wewnętrzną `B`)
c) Jest to niepoprawny kod (błąd występuje w metodzie `main`)
d) Jest to poprawny kod

27. Co można powiedzieć o poniższym kodzie?

```java
package pA;
import pB.B;

public class A {
    private B b;

    A(B b) {
        this.b = b;
    }
}
```

```java
package pB;

public class B {
    B(){}
}
```

a) Jest to niepoprawny kod (w klasie `A` nie można tak zaimportować klasy `B`)
b) Jest to niepoprawny kod (pojawi się informacja o ograniczonym dostępie)
c) Jest to poprawny kod, lecz utworzenie obiektu klasy `A` w innym źródle kodu nie będzie możliwe
d) Jest to poprawny kod, a utworzenie obiektu klasy `A` w innym źródle kodu będzie możliwe

28. Co można powiedzieć o poniższym kodzie?

```
public class A {
    enum E {
        A, B, C() {
            void m() { System.out.print("C"); }
        };
        void m() { System.out.print("X"); }
    };
    public static void main(String[] args) {
        for (E e : E.values())
            e.m();
    }
}
```

a) Jest to niepoprawny kod (w deklaracji typu wyliczeniowego nie można tak powielać nazwy metody `m`)
b) Jest to niepoprawny kod (nie można deklarować osobnego konstruktora jak dla elementu `C`)
c) Jest to niepoprawny kod (źle zadeklarowano pętlę `for`)
d) Jest to poprawny kod

29. Która z metod klasy `DatagramPacket` służy do pobrania numeru portu?

a) `port()`
b) `findPort()`
c) `getPort()`
d) `recievePort()`

30. Co można powiedzieć o poniższym kodzie?

```
public class A  {
    void m(int a) { }
    int m(String s){ return Integer.parseInt(s);}
    String m(int a){ return Integer.toString(a);}
}
```

a) Jest to niepoprawny kod (złe przeciążenie metody `m`)
b) Jest to niepoprawny kod (nieobsłużony wyjątek `NumberFormatException`)
c) Jest to niepoprawny kod (zła konwersja z `int` do `String`)
d) Jest to poprawny kod