Zadanie 1/ operatory

Jaki będzie wynik działania programu?

```
class Demo {
    public static void main(String[] args){
        int i = 3;
        i++;
        System.out.println(i);
        ++i;
        System.out.println(i);
        System.out.println(++i);
        System.out.println(i++);
        System.out.println(i);
    }
}
```

Zadanie 2/ typy danych, operatory

Jaki będzie wynik działania programu w zaznaczonych liniach? Czy program się skompiluje?

```
public static void funCasting(){
        int x = 128;
        Integer xObjInt = x;
        Double xObjDouble = xObjInt.doubleValue();
        byte xByte = (byte) x;
        System.out.println(xObjInt); //1
        System.out.println(xObjDouble); //2
        System.out.println(xByte); //3

        Double y = 2.623;
        Integer yObjInt = y.intValue();
        Integer yMathInt = (int)Math.round(y);
        System.out.println(yObjInt.equals(yMathInt)); //4

       Integer i = 65;
        char c = (char) i.intValue();
        System.out.println(c); //5

        String s = "\\\u0041";
        int intForS = 0041;
        char charForS = 0041;
        String s2 = "\\u" + intForS;
        String s3 = "\\u" + charForS;
        System.out.println(s); //6
        System.out.println(s2); //7
        System.out.println(s3);//8

        String s4 = 5 + "5";
        String s5 = 5 + 5 + "5";
        String s6 = "5" + 5 + 5;
        System.out.println(s4); //9
        System.out.println(s5); //10
        System.out.println(s6); //11

      Double d1 = 5d;
        Double d2 = 0d;
        Double div = d1/d2;
        System.out.println(div); //*12
}
```

Zadanie 3/ parametry, kolekcje

Jaki będzie wynik działania programu? Czy program się skompiluje?

*Jaki będzie wynik działania programu, gdy w funkcji main wyrzucimy new ArrayList<>?

```
public static void fun(List<String> list){
        list.add("three");
        list.add("four");
}
public static void main(String[] args) {
        List<String> list = new ArrayList<>(Arrays.asList("one", "two"));
        fun(list);
        System.out.println(list);
}
```

Zadanie 4/ String, StringBuilder

Jakie będą wyniki działania programu dla poszczególnych linii kodu?

```
package com.sda.main;

public class Main {
    public static void main(String[] args) {
        String s = "text";
        s.toUpperCase().concat("x2");
        System.out.println("1: " + s); //1

        StringBuilder sb = new StringBuilder(s);
        sb.append("x1").append("x2").append("x3");
        System.out.println("2: " + sb); //2

        String s1 = "abc";
        String s2 = "abc";
        String s3 = new String("abc");
        String s4 = "ABC".toLowerCase();
        String s5 = "a" + "b" + "c";

        System.out.println("3: " + (s1 == s2)); //3
        System.out.println("4: " + s1.equals(s2)); //4
        System.out.println("5: " + (s1 == s3)); //5
        System.out.println("6: " + s1.equals(s3)); //6
        System.out.println("7: " + (s1 == s4)); //7
        System.out.println("8: " + s1.equals(s4)); //8
        System.out.println("9: " + (s1 == s5)); //9
    }
}
```

Zadanie 5/ składnia

Jaki będzie wynik działania programu? Czy program się skompiluje?

```
import java.util.LinkedList;
import java.util.List;
package com.sda.main;

public class Main {
    public static void main(String[] args) {
        List<Integer> ints = new LinkedList<>();
        ints.add(new Double(2.1).intValue());
        System.out.println(ints);
    }
}
```

Zadanie 6/ obiekty, pola, konstruktory

Jaki będzie wynik działania programu? Jaki będzie wynik działania programu po zakomentowaniu sekcji oznaczonej jako pierwsza? Czy w jednym i drugim przypadku program się skompiluje?

```
package com.sda.main;

class Snake{
    int length;
    Snake(int length){
        length = length;
    }
}

public class Main {
    public static void main(String[] args) {
        Snake snake;
        // 1
        snake = new Snake();
        snake.length = 5;
        System.out.println(snake.length);

        // 2
        snake = new Snake(10);
        System.out.println(snake.length);
    }
}
```

Zadanie 7/ obiekty, pola, konstruktory, wzorce projektowe, clean code

Mamy daną następującą klasę:

```
public class Person {
    String firstName;
    String middleName;
    String familyName;
    String fatherFirstName;
    String motherFirstName;
    Integer height;
    String nationality;
    String eyeColor;
    LocalDate birthday;
    Boolean isEmployed;
    Boolean isFemale;
    Boolean isHomeOwner;
    Boolean isInsured;
    String state;
    String city;
    String streetAddress;
    String postalCode;

    public Person(String firstName, String middleName, String familyName, String
fatherFirstName, String motherFirstName, Integer height, String nationality,
String eyeColor, LocalDate birthday, Boolean isEmployed, Boolean isFemale, Boolean
isHomeOwner, Boolean isInsured, String state, String city, String streetAddress,
String postalCode) {
        this.firstName = firstName;
        this.middleName = middleName;
        //...skipped for brevity
    }
}
```

Przyjrzyj się uważnie powyższej klasie. Jakich zmian należy dokonać, aby ją poprawić, mając na uwadze bezpieczeństwo, czytelność oraz praktyczność zastosowania takiej klasy?

Tak złożony konstruktor może sprawiać problemy. Czy jesteś w stanie przewidzieć jakie? W jaki sposób poradzić sobie z problemem obszernego konstruktora?

Na podstawie powyższych przemyśleń zaimplementuj rozwiązanie tych problemów.

Zadanie 8/ tablice

Jaki będzie wynik działania programu? Czy program się skompiluje? Jaki byłby wynik działania programu osobno dla pierwszej i drugiej sekcji kodu?

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

        // 1
        a = new int[3][3];
        for (int i = a.length - 1; i > 0; i--) {
            for (int j = 0; j < a[i].length; j++){
                System.out.print(a[i][j] + " ");
            }
            System.out.println();
        }

        // 2
        b = new int[][]{{1, 2},{3,4,5}};
        System.out.println(b[0][1]);
    }
}
```

Zadanie 9/ tablice

Mamy daną następującą tablicę int:

a) `int a[];`
b) `int a[][];`

Jaki jest najszybszy sposób, żeby wypełnić taką tablicę zerami?

W przypadku b – jak to zrobić dla drugiego wymiaru, żeby każda kolejna tablica miała inny rozmiar?

Zadanie 10/ tablice, algorytmy

Sudoku jest grą polegającą na uzupełnieniu cyframi w przedziale 1-9 siatki o wymiarze 9x9 podzielonej na 9 kwadratów o wymiarach 3x3, tak aby:

1. wszystkie pola zostały wypełnione cyframi,
2. w każdym rzędzie cyfry nie powtarzały się,
3. w każdej kolumnie cyfry nie powtarzały się,
4. w każdym kwadracie cyfry nie powtarzały się.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Korzystając z techniki TDD napisz testy i algorytm, który sprawdzi czy ukończona siatka sudoku została wypełniona poprawnie. Spróbuj znaleźć jak najbardziej optymalne rozwiązanie.

Zastanów się, jakie są możliwe przypadki testowe takiego algorytmu. Zastosuj testy parametryzowane.

W jaki sposób wyliczyć, w którym z 9 kwadratów znajduje się aktualnie pobrana z tablicy liczba? W jaki sposób najlepiej sprawdzać czy liczby nie powtarzają się w rzędzie, kolumnie czy kwadracie?

Jako dane wejściowe przyjmij tablicę dwuwymiarową int s = new int[9][9].

Dla utrudnienia możesz użyć testów parametryzowanych.

Zadanie 11/ kolekcje, Integer, int, tablice

Jaki będzie wynik działania programu? Czy program się skompiluje? Jeśli program skompilowałby się, w jaki sposób można wyciągnąć elementy z obu list?

```
public class Main {

    private static void generateIntsLists(){
        List integers = Arrays.asList(new Integer[]{1, 2, 3});
        List ints = Arrays.asList(new int[]{1, 2, 3});
        System.out.println(integers.size()==ints.size());
    }

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

Zadanie 12/ statements

Jaki będzie wynik działania programu? Czy program się skompiluje?

```
package com.sda.main;

public class Main {
    public static void main(String[] args) {
        fun(5);
        fun(2);
    }

    private static void fun(int x){
        switch (x){
            case 1: case 6:
                System.out.println("Apple");
                break;
            case 2:
                System.out.println("Banana");
            case 3:
                System.out.println("Orange");
                break;
            default:
                System.out.println("Kiwi");
        }
    }
}
```

Zadanie 13/ pętle

Jaki będzie wynik działania programu? Czy program się skompiluje?

```
package com.sda.main;
import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        fun();
    }

    private static void fun(){
        List<String> items = new ArrayList<>();
        items.add("Pen");
        items.add("Pencil");
        items.add("Box");
        for (String i : items) {
            if (i.indexOf("P") == 0) {
                continue;
                System.out.print("beginds with p: " + i);
            } else {
                System.out.print(i+" ");
            }
        }
    }
}
```

Zadanie 14/ OOP, pętle, statements, algorytmy

Jaki będzie wynik działania programu? Czy program się skompiluje?

Co oblicza funkcja fun()? W funkcji fun() jest pewien błąd. Jak go naprawić? Jaki wynik zwróci po naprawie błędu dla wartości 4?

```
public class Main {
    public static void main(String[] args) {
        System.out.println(fun(4));
    }

    private int fun(int x){
        return x*fun(x-1);
    }
}
```

Zadanie 15/ funkcje, konstruktory

Jaki będzie wynik działania programu? Czy program się skompiluje?

```
package com.sda;

public class Ex {
    private static int x =1;

    public Ex(){
        System.out.println("constructor");
        x=0;
    }

    static {
        System.out.println("first" + x++);
    }

    private static void fun(){
        System.out.println("fun" + x);
    }

    {
        System.out.println("second" + ++x);
    }

    public static void main(String[] args) {
        Ex ex = new Ex();
        fun();
    }
}
```

Zadanie 16/ OOP, streamy, interfejsy, klasy abstrakcyjne, dziedziczenie, polimorfizm, kompozycja, logger, immutable

Stosując technikę TDD zaimplementuj prosty program służący do zarządzania powierzchnią biurową według poniższych wymagań:

1. Stwórz interfejs `Measurable`. Interfejs powinien zawierać:
    a. Metodę abstrakcyjną `Double area(),`
    b. Metodę defaultową `String show()`, która zwróci niespecyficzny opis obiektu, np. `„Not labelled yet.”.`
2. Stwórz klasę abstrakcyjną `Furniture`. Klasa powinna implementować `Measurable` i zawierać pole prywatne i finalne `Double price`,
3. Stwórz klasę immutable `Table`. Klasa powinna dziedziczyć z `Furniture` i zawierać pola `Double length`, `Double width` i `Double height`,
4. Stwórz klasę `SquareCabinet`. Klasa powinna dziedziczyć z `Furniture` i zawierać pole `Double length`,
5. Stwórz klasę `RectangularRoom`. Klasa powinna implementować `Measurable` i zawierać pola `Double width`, `Double length` i `List<Furniture> furnitures`,
6. Stwórz wyjątek `ExceededRoomAreaException`.
7. Stwórz klasę menagera powierzchni biurowej `InteriorManager`. Klasa powinna:
    a. Zawierać listę wszystkich dostępnych zasobów (mebli i pokoi) o nazwie `resources`,
    b. Do wypisywania tekstu posługiwać się klasą loggera z `java.util.logging.Logger` (podstawowa instrukcja używania loggera na końcu zadania),
    c. Implementować funkcję `void addRoomWithFurniture`, która:
        i. Przyjmie jako parametr jeden obiekt pokoju typu `RectangularRoom` i dowolną liczbę obiektów mebli typu `Furniture` (nie używaj `List<Furniture>`),
        ii. Jeśli powierzchnia mebli jest mniejsza lub równa powierzchni pokoju
            I. przypisze meble do pokoju,
            II. doda wszystkie zasoby (meble i pokój) do listy zasobów `resources`,
            III. zaloguje sumaryczną cenę dodanych mebli.
        iii. Jeśli jest większa – wyrzuci wyjątek ExceededRoomAreaException.
    d. Implementować funkcję `Double countRoomsTotalArea`, która zwróci sumaryczną powierzchnię wszystkich pokoi z listy zasobów `resources`.
    e. Implementować funkcję `void showAllRecources`, która zaloguje wszystkie zasoby z listy `resources`.

Wskazówki:

- Twórz tylko potrzebne metody.
- Używaj streamów.
- Używaj loggera.
- Staraj się pisać jak najmniej złożony kod.
- Pamiętaj o enkapsulacji.
- Korzystaj z dobrodziejstw polimorfizmu i dziedziczenia.
- Zapoznaj się z loggerem na przyszłość. ;)

Podstawowe używanie Loggera(wystarczające dla tego zadania):

```
private Logger logger = Logger.getLogger(InteriorManager.class.getName());
```

```
public void fun(){
	logger.info("message");
}
```

Dla chętnych – zapis do pliku:

```
logger.addHandler(new FileHandler(„/ścieżka/do/pliku.log”));
```