Proszę korzystać z dobrych praktyk programowania oraz pilnować enkapsulacji w programach niech pola mają modyfikatory dostępów najbardziej rygorystyczne jak się da.

1.

Utwórz metodę przyjmującą dowolną ilość argumentów typu String oraz

pojedynczy argument typu int (pamiętaj o zachowaniu odpowiedniej kolejności w deklaracji -> patrz

slajdy) i zwracającą wynik w postaci String.

Niech argument typu int definiuje co n-te elementy, które zostaną usunięte np. przy użyciu

argumentu int z wartością 2 spodziewamy się, że metoda usunie co drugie słowo z zadanego tekstu,

tj :

„Ala", „nie", „ma", „test" , kota i", „psa", „papugę" 2 -> „Ala ma kota i papugę" // 2-gie i czwarte

argumenty zostały usunięte

2.

Odpowiedz na pytania na podstawie kodu:

```
public class ClassA {
    public void methodOne(int i) {
    }
    public void methodTwo(int i) {
    }
    public static void methodThree(int i) {
    }
    public static void methodFour(int i) {
    }
}

public class ClassB extends ClassA {
    public static void methodOne(int i) {
    }
    public void methodTwo(int i) {
    }
    public void methodThree(int i) {
    }
    public static void methodFour(int i) {
    }
}
```

1. Które z metod nadpisują metodę klasy nadrzędnej?
2. Które z metod przysłaniają metodę klasy nadrzędnej?
3. Co robią pozostałe metody?

3.

Stwórz klasy A, B, C, D, E. Klasa B dziedziczy po A, C po B, D po C, E po D. Niech każda z tych klas ma metodę present(String phrase); Niech klasa A w metodzie present(String phrase) ma drukować argument podany do metody. Z kolei każda z podklas (B,C,D,E) ma w swojej metodzie wywoływać metodę klasy nad nią. (Czyli gdy wywołamy metodę w E, to ma ona wywołać metodę w D, ona zaś w C, która wywoła w B, która wywoła w A, która w końcu wydrukuje co ma wydrukować.)

4.

Napisz metodę która:

1. wiersze, oraz kolumny muszą mieć długość niemniejszą, niż 3.

2. z sumuje elementy dwuwymiarowej tablicy.

b) Napisz metodę, która sprawdzi czy nasza tablica dwuwymiarowa ma unikalne elementy, jeżeli tak zwróć true, jeżeli nie zwróć false.

5. Napisz klasę pasażer, która ma pola: imię ,nazwisko, wiek, oraz przyjmuje klasę enum bilet.

a) bilet ma w sobie 2 pola ULGOWY oraz NORMALNY

b)Utwórz metodę jedź oraz sprawdź czy pasażer może jeździć na bilecie ulgowym (jeżeli ma 26 lat lub więcej to nie może), jeżeli ma mniej niż 26 i jeździ na bilecie normalnym to poinformuj pasażera, że jeżeli dalej kontynuuje naukę to może korzystać z biletów normalnych

c) stwórz 3 instancję obiektu pasażer

6. Zaimplementuj program:

a) Napisz metodę, która przyjmuje argument int

b) Rozbij liczbę jeżeli jest 3 cyfrowa na : setki, dziesiątki, jedności, 2 cyfrowa: dziesiątki, jedności, 1 cyfrowa na same jedności.

Przykład:

Jeżeli mamy liczbę np. 145 to sprawdź czy 1! + 4! + 5! Zwróci 145 , jeżeli tak zwróć true, jeżeli nie to zwróć false.

Przykład:

Jeżeli mamy liczbę 2 to zwróci true, gdyż 2! == 2

7. Czym jest rekurencja?
a) Jak działa ?

b) czy jest wydajna ?

c) Zaimplementuj jakiś przykład w kodzie.

8.* Napisz implementację arraylist oraz skorzystaj z niej w psvm:

Niech ma pola:

a) Tablice obiektów, oraz aktualny rozmiar
b) Niech ma konstruktor, który automatycznie ustawia rozmiar tablicy Object na 10
c) Niech ma metody:
    a) Add (przyjmuję argument object) dodaje element do tablicy, oraz sprawdza czy możemy dodać kolejny element, jeżeli nie to rozszerzamy rozmiar naszej tablicy razy 2
    b) Get (przyjmuje argument index) oraz zwraca element w tablicy tab[index], jeżeli nie ma danego indeksu w tablicy rzuć wyjątkiem IndexOutOfBoundsException.
    c) Remove (przyjmuje argument index) oraz usuwa element z tablicy tab[index], jeżeli nie ma danego indeksu w tablicy rzuć wyjątkiem IndexOutOfBoundsException.
    d) Metodę increase size:

```
private void IncreaseSize(){
    mysStore = Arrays.copyOf(mysStore, newLength: mysStore.length*2);

}
```

Gdzie mystore jest tablicą obiektów