Zbiór zadań - Java

Piotr Jastrzębski

2023-07-02

Spis treści

Zbiór zadań - Java

Tu będzie zbiór zadań z programowania w języku Java. Inspiracją było zebranie zadań powstałych w trakcie prowadzenia zajęć dydaktycznych realizowanych na Wydziale Matematyki i Informatyki Uniwersytetu Warmińsko-Mazurskiego w Olsztynie.

Cześć I.

Wprowadzenie do języka Java

1. Instrukcje wejścia/wyjścia

1. Napisz prostą aplikację kalkulatora tekstowego, która przyjmuje dwa liczby od użytkownika jako wejście i wykonuje podstawowe operacje matematyczne (dodawanie, odejmowanie, mnożenie, dzielenie). Wyświetl wyniki na ekranie.

Cześć II.

Programowanie obiektowe

2. Konstruktor

1. Stwórz klasę `Samochod` zawierającą prywatne pola: `marka`, `model`, `rokProdukcji`, `przebieg` oraz `kolor`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) dla wszystkich pól. Następnie dodaj metodę `wyswietlInformacje()`, która wyświetla wszystkie informacje o samochodzie.

2. Stwórz klasę `Osoba` z prywatnymi polami: `imie`, `nazwisko`, `wiek`, `adres`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `przedstawSie()`, która zwraca łańcuch znaków z informacjami o osobie.

3. Stwórz klasę `Ksiazka` z prywatnymi polami: `tytul`, `autor`, `rokWydania`, `wydawnictwo` oraz `liczbaStron`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `pokazInformacje()`, która wyświetla informacje o książce.

4. Stwórz klasę `Punkt2D` z prywatnymi polami `x` i `y`, reprezentującymi współrzędne punktu na płaszczyźnie. Dodaj konstruktor, który przyjmuje współrzędne jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `odleglosc(Punkt2D innyPunkt)`, która oblicza odległość między dwoma punktami na płaszczyźnie.

5. Stwórz klasę `Prostokat` z prywatnymi polami `szerokosc` i `wysokosc`. Dodaj konstruktor, który przyjmuje długości boków jako argumenty.

Dodaj metody dostępowe (gettery i settery) oraz metody `pole()` i `obwod()`, które obliczają pole powierzchni i obwód prostokąta.

6. Stwórz klasę `Kolo` z prywatnym polem `promien`. Dodaj konstruktor, który przyjmuje promień jako argument. Dodaj metody dostępowe (gettery i settery) oraz metody `pole()` i `obwod()`, które obliczają pole powierzchni i obwód koła.

7. Stwórz klasę `Student` z prywatnymi polami: `imie`, `nazwisko`, `numerIndeksu`, `rokStudiow` oraz `sredniaOcen`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `pokazInformacje()`, która wyświetla informacje o studencie.

8. Stwórz klasę `Pracownik` z prywatnymi polami: `imie`, `nazwisko`, `stanowisko`, `wiek` oraz `placa`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `pokazInformacje()`, która wyświetla informacje o pracowniku.

9. Stwórz klasę `KontoBankowe` z prywatnymi polami: `numerKonta`, `wlasciciel`, `saldo` oraz `typKonta`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metody `wplac(double kwota)` i `wyplac(double kwota)`, które odpowiednio dodają lub odejmują kwotę od salda konta.

10. Stwórz klasę `Telewizor` z prywatnymi polami: `marka`, `przekatnaEkranu`, `rozdzielczosc`, `czySmartTV` oraz `cena`. Dodaj konstruktor, który przyjmuje wszystkie pola jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `pokazInformacje()`, która wyświetla informacje o telewizorze.

11. Stwórz klasę `DziennikOcen` z prywatnymi polami: `imie`, `nazwisko` oraz `oceny` (jako `ArrayList` typu `int`). Dodaj konstruktor, który przyjmuje imię i nazwisko jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metody `dodajOcene(int ocena)` i

`usunOcene(int indeks)`, które odpowiednio dodają lub usuwają ocenę z listy ocen. Dodaj również metodę `sredniaOcen()` do obliczania średniej ocen.

12. Stwórz klasę `HistoriaTemperatur` z prywatnym polem temperatury (jako `ArrayList` typu `double`). Dodaj konstruktor domyślny. Dodaj metody dostępowe (gettery i settery) oraz metody `dodajTemperature(double temperatura)` i `usunTemperature(int indeks)`, które odpowiednio dodają lub usuwają temperaturę z listy temperatur. Dodaj również metodę `sredniaTemperatur()` do obliczania średniej temperatur.

13. Stwórz klasę `WynikiTestow` z prywatnymi polami: `imie`, `nazwisko` oraz `wyniki` (jako tablica typu `int`). Dodaj konstruktor, który przyjmuje imię, nazwisko oraz rozmiar tablicy jako argumenty. Dodaj metody dostępowe (gettery i settery) oraz metodę `dodajWynik(int indeks, int wynik)`, która dodaje wynik testu na podanym indeksie. Dodaj również metodę `sredniWynik()` do obliczania średniego wyniku.

14. Stwórz klasę `ZarzadcaZadan` z prywatnym polem `priorytetyZadan` (jako `ArrayList` typu `int`). Dodaj konstruktor domyślny. Dodaj metody dostępowe (gettery i settery) oraz metody `dodajPriorytet(int priorytet)` i `usunPriorytet(int indeks)`, które odpowiednio dodają lub usuwają priorytet z listy priorytetów. Dodaj również metodę `najwyzszyPriorytet()` do znajdowania najwyższego priorytetu.

15. Stwórz klasę `Magazyn` z prywatnym polem `iloscProduktow` (jako tablica typu `int`). Dodaj konstruktor, który przyjmuje rozmiar tablicy jako argument. Dodaj metody dostępowe (gettery i settery) oraz metodę `dodajProdukty(int indeks, int ilosc)`, która dodaje określoną ilość produktów na podanym indeksie. Dodaj również metodę `sumaProduktow()` do obliczania sumy wszystkich produktów w magazynie.

3. Dziedziczenie

1. Utwórz klasę `Pojazd` z polami `marka`, `model` i `rokProdukcji`. Utwórz klasy `Samochod` i `Motocykl`, które dziedziczą po klasie `Pojazd`. Klasa `Samochod` powinna mieć dodatkowe pole `liczbaDrzwi`, a klasa `Motocykl` pole `pojemnoscSilnika`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

2. Utwórz klasę `Pracownik` z polami `imie`, `nazwisko` i `placa`. Utwórz klasy `Programista` i `Tester`, które dziedziczą po klasie `Pracownik`. Klasa `Programista` powinna mieć dodatkowe pole `jezykProgramowania`, a klasa `Tester` pole `typTestowania`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

3. Utwórz klasę `Nieruchomosc` z polami `adres`, `metraż` i `cena`. Utwórz klasy `Dom` i `Mieszkanie`, które dziedziczą po klasie `Nieruchomosc`. Klasa `Dom` powinna mieć dodatkowe pole `liczbaPieter`, a klasa `Mieszkanie` pole `numerPietra`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

4. Utwórz klasę `GraPlanszowa` z polami `nazwaGry`, `minLiczbaGraczy`, `maxLiczbaGraczy` oraz `zasadyGry` (jako ArrayList typu `String`). Utwórz klasy `GraEdukacyjna` i `GraStrategiczna`, które dziedziczą

po klasie `GraPlanszowa`. Klasa `GraEdukacyjna` powinna mieć dodatkowe pole `przedmiot`, a klasa `GraStrategiczna` pole `czasTrwania`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

5. Utwórz klasę `Druzyna` z polami `nazwa`, `miasto` oraz `punkty` (jako ArrayList typu `Integer`). Utwórz klasy `DruzynaPilkarska` i `DruzynaSiatkarska`, które dziedziczą po klasie `Druzyna`. Klasa `DruzynaPilkarska` powinna mieć dodatkowe pole `pozycjaWRankingu`, a klasa `DruzynaSiatkarska` pole `liczbaZwyciestw`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

6. Utwórz klasę `Komputer` z polami `producent`, `model` oraz `cenyCzesci` (jako ArrayList typu `Double`). Utwórz klasy `Laptop` i `Stacjonarny`, które dziedziczą po klasie `Komputer`. Klasa `Laptop` powinna mieć dodatkowe pole `waga`, a klasa `Stacjonarny` pole `obudowa`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

7. Utwórz klasę `AlbumMuzyczny` z polami `tytul`, `artysta` oraz `oceny` (jako ArrayList typu `Integer`). Utwórz klasy `AlbumRockowy` i `AlbumJazzowy`, które dziedziczą po klasie `AlbumMuzyczny`. Klasa `AlbumRockowy` powinna mieć dodatkowe pole `gatunekRocka`, a klasa `AlbumJazzowy` pole `gatunekJazzu`. Dodaj konstruktory, metody gettery i settery, metodę `toString()` oraz `equals()` dla każdej z klas. Napisz program testujący zdefiniowane klasy i metody.

8. Zdefiniuj abstrakcyjną klasę `NarzedziePracy` z polami `nazwa` typu `String` oraz `rokProdukcji` typu `java.time.LocalDate`. Dodaj metodę abstrakcyjną `uzyj()`, która będzie symulować użycie narzędzia. Następnie zdefiniuj klasy `Mlotek`, `Srubokret` i `Pila`, które dziedziczą po klasie `NarzedziePracy` i implementują metodę `uzyj()`. Stwórz

listę tablicową odpowiednich 5 obiektów i wywołaj dla nich napisaną metodę.

9. Zdefiniuj abstrakcyjną klasę `GrafikaKomputerowa` z polami `szerokosc`, `wysokosc` typu `int` oraz `nazwaPliku` typu `String`. Dodaj abstrakcyjne metody `wczytajPlik()` i `zapiszPlik()`. Następnie zdefiniuj klasy `Bitmapa` i `Wektor`, które dziedziczą po klasie `GrafikaKomputerowa` i implementują metody `wczytajPlik()` oraz `zapiszPlik()`. Stwórz listę tablicową odpowiednich 5 obiektów i wywołaj dla nich napisaną metodę.

10. Zdefiniuj abstrakcyjną klasę `UrządzenieElektroniczne` z polami `producent` typu `String`, `model` typu `String` oraz `rokProdukcji` typu `java.time.LocalDate`. Dodaj abstrakcyjne metody `włącz()` i `wyłącz()`. Następnie zdefiniuj klasy `Smartfon`, `Telewizor` i `Laptop`, które dziedziczą po klasie `UrządzenieElektroniczne` i implementują metody `włącz()` oraz `wyłącz()`. Stwórz listę tablicową odpowiednich 5 obiektów i wywołaj dla nich napisaną metodę.

Bibliografia i inne zbiory zadań

Jan, Kozak. b.d. *Materiały do ćwiczeń*. http://www.jkozak.pl/przedmioty/podstawy-i-jezyki-programowania/materialy-do-cwiczen/.

Rychlicki, Wiesław. 2012. *Programowanie w języku Java. Zbiór zadań z (p)odpowiedziami*. Helion.