Języki i metody programowania – Java INF302W Wykład 2 (część 1)

Autor

Dr inż. Zofia Kruczkiewicz

Struktura wykładu

1. **Identyfikacja złożonych danych reprezentowanych przez klasy powiązane dziedziczeniem i polimorfizmem podczas opracowania koncepcji programu obiektowego. Dziedziczenie (polimorfizm) – przedefiniowanie metod, słowo kluczowe super, tablice z obiektowymi elementami, klasy usługowe (R-1, EL-1).**
2. **Identyfikacja złożonych danych opartych wspólności i zmienności cech podczas opracowania koncepcji programu obiektowego. Interfejsy i pakiety, implementacja interfejsów, klasy usługowe cd (1).**

1. Tablice obiektów

Tablica w Javie jest obiektem.

- **Deklarowanie tablicy**

String [] nazwy; //równoważne deklaracje zmiennej tablicowej
String nazwy []; // czyli referencji do obiektu tablicy, deklarujące elementy obiektowe
int liczby[]; //zamienna tablicowa deklarująca elementy nieobiektowe

- **Tworzenie obiektu tablicowego**

int liczby []= **new** **int** [10];
// utworzono tablicę 10 elementów typu **int**
String nazwy []= **new** String[10];
// utworzono tablicę 10 referencji typu String, należy dla każdego elementu tablicy przydzielić pamięć

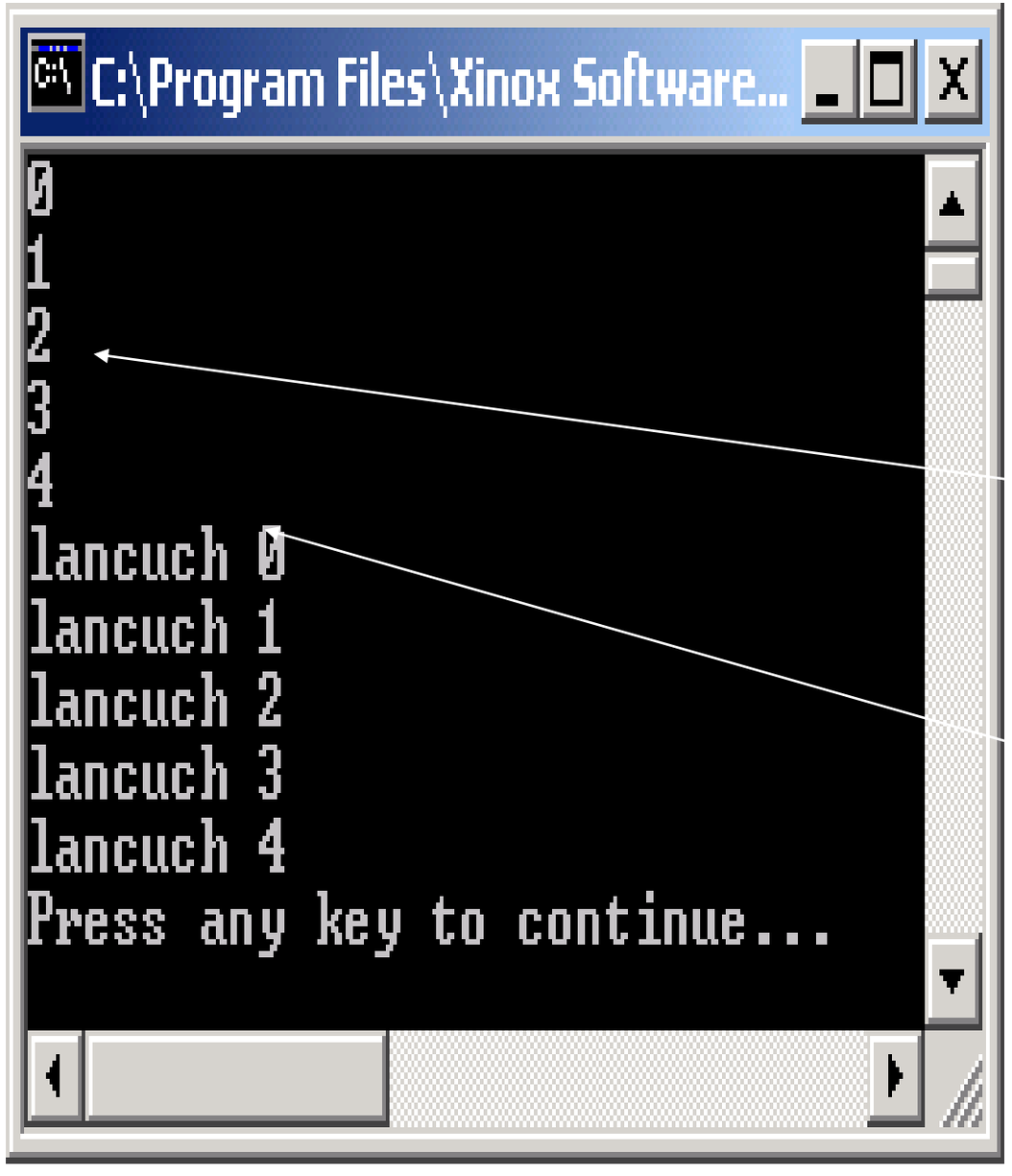
- **Przydział pamięci na elementy obiektowe tablicy**

np. nazwy[0]= **new** String(”Pewien wiersz”);
//przydzielono pamięć na obiekt typu String w pierwszym elemencie tablicy (liczby[0])

- **Pobranie rozmiaru tablicy**

nazwy.length

Przykład 1




```
import java.lang.*;
public class Tablice
{
   public static void main(String args[])
     { final int N=5;

        int liczby [] = new int [N];
        for (int i=0; i<liczby.length; i++)
              {
                   liczby[i]= i;
                 System.out.println(liczby[i]);
              }

         String nazwy[]=new String[N];
           for (int i=0; i<nazwy.length; i++)
             {
                   nazwy[i]=new String("lancuch "+i);
                 System.out.println(nazwy[i]);
              }
      }
}
```

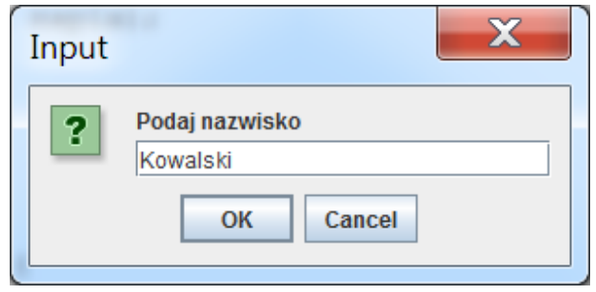
2. Definicja elementu tablicy

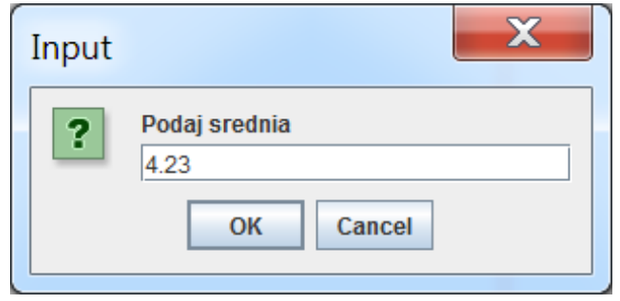
Przykład 2

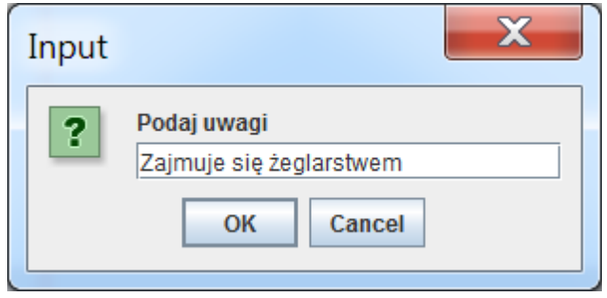
```
import javax.swing.*;

class Osoba1                        //element tablicy wykorzystany w kolejnych przykladach
 {
   String nazwisko;
   float srednia;
   String uwagi;
   static int numer = 0;            //atrybut klasowy, ktorego wartosc jest wykorzystana do
                                     //wyznaczania liczby obiektow typu Osoba1

 public void Nadaj_dane(String []dane)
  { Inicjuj();                      //wyznaczenie liczby tworzonych obiektów typu Osoba1
    nazwisko=dane[0];
    srednia= Float.parseFloat(dane[1]);
    uwagi=dane[2];
  }
public String Podaj_dane()
 { String napis="";
    napis+="\n Nazwisko: "+nazwisko;
    napis+="\n Średnia: "+srednia;
    napis+="\n Uwagi: "+uwagi;
    napis+="\n Liczba osób jest równa "+numer;
   return napis; }
public void Inicjuj()    { numer++; }
}
```

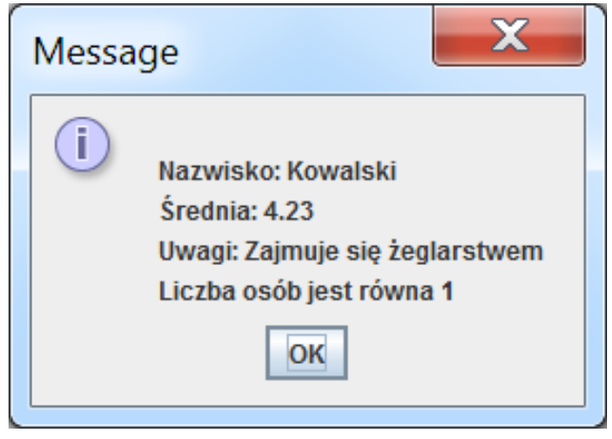
```
class GUI1{
public String[] Wstaw()
  {  String S[]=new String[3];
     S[0] = JOptionPane.showInputDialog(null, "Podaj nazwisko");
     S[1] = JOptionPane.showInputDialog(null, "Podaj srednia");
     S[2] =JOptionPane.showInputDialog(null, "Podaj uwagi");
     return S;
  }

public void Wyswietl(String napis)
  {  JOptionPane.showMessageDialog(null, napis); }
 }
```

```
public class Cala
 {
  static public  void main(String args[])
   {
     Osoba1 os1 = new Osoba1();
     GUI1 gui1= new GUI1();
     os1.Nadaj_dane(gui1.Wstaw());
     gui1.Wyswietl(os1.Podaj_dane());
     System.exit(0);
   }
 }
```







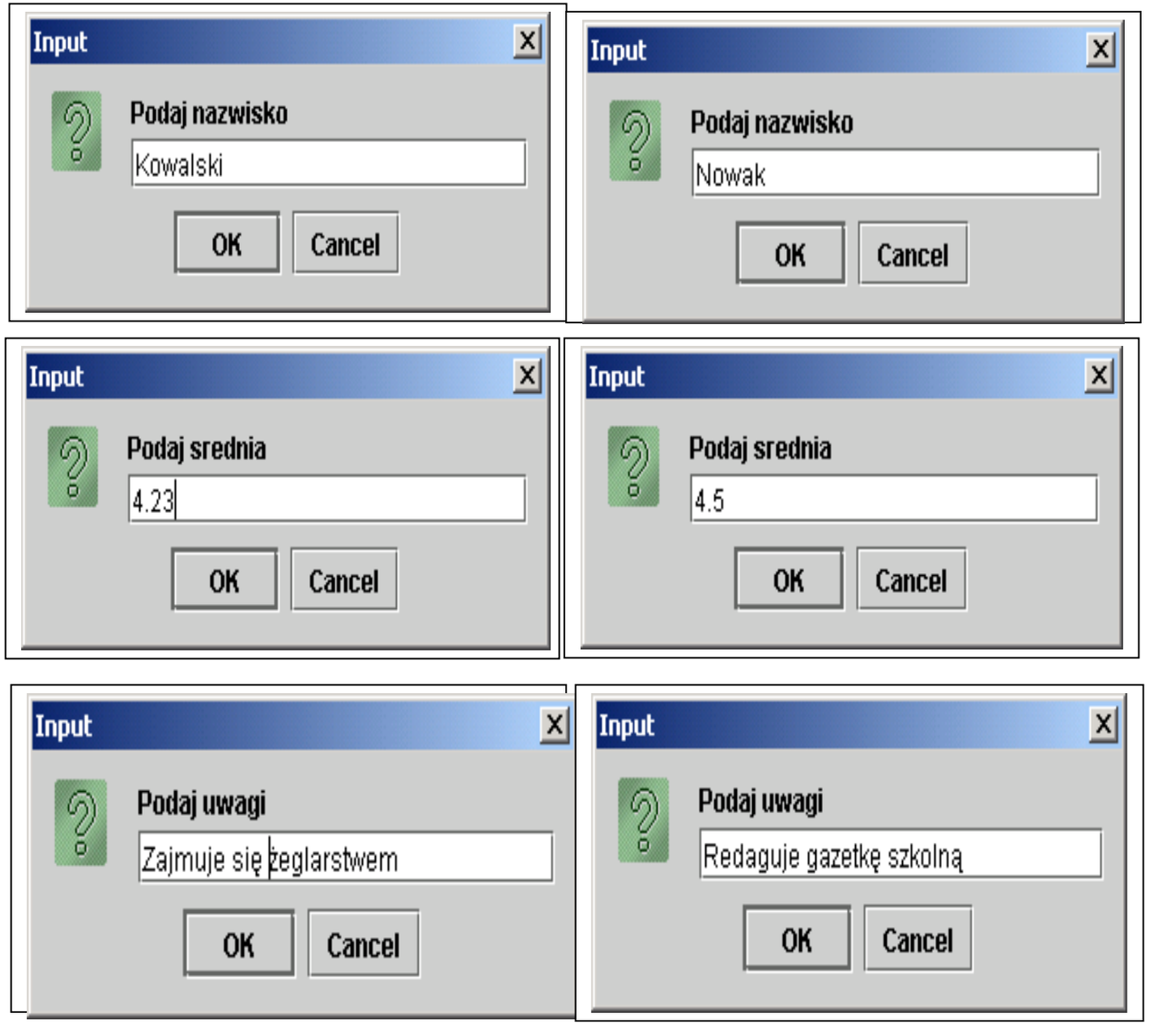


3. Tablica obiektów – operacje wstawiania i wyświetlania

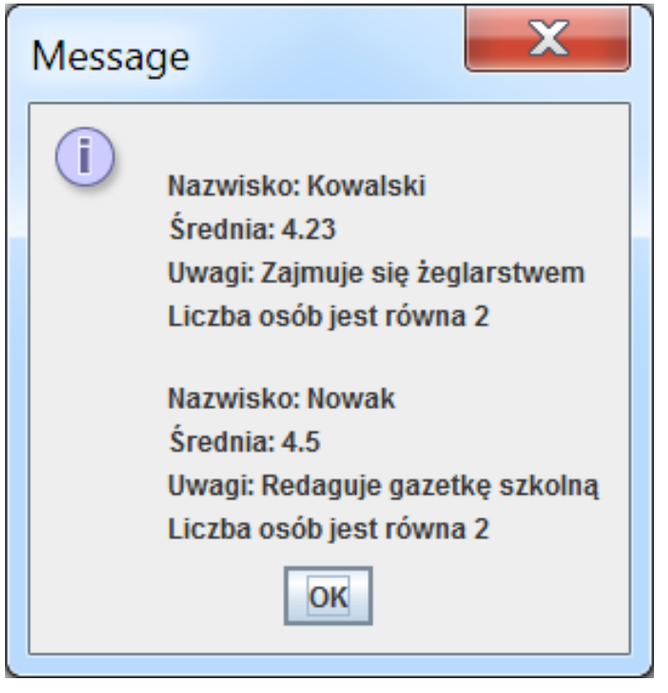
Przykład 3 -

```
import javax.swing.*;

class Osoba1
   {// kod klasy Osoba1 z przykładu 2
   }
class GUI1
   {//kod klasy GUI1 z przykladu 2
   }
 class Tablica {

   Osoba1 osoby[]=new Osoba1[N];
   static int N = 2;
   int ile;                                  //składowa do wyznaczania liczby elementów w tablicy

   public void Dodaj_osobe(String[] dane) {
      if (Pelna())
         return;                             //nie mozna wstawic wiecej niz 2 osoby do tablicy
      Osoba1 nowa = new Osoba1();
      nowa.Nadaj_dane(dane);
      ile++;
      osoby[ile - 1] = nowa;                 //dodanie obiektu typu Osoba1 do kolejnego elementu tablicy
   }
   boolean Pelna() {
      return ile == N; }                     //wartosc wyrazenia true oznacza, ze tablica jest pelna
```

```
public String Podaj_dane_tablicy() {
        String dane_tablicy = "";
        for (Osoba1 osoba : osoby) { //pętla działa na wszystkich elementach tablicy, czyli
                                     //równej osoby.length
            dane_tablicy += osoba.Podaj_dane() + "\n";  //wykonanie wielowierszowego
        }                                                // łańcucha danych osób
        return dane_tablicy;
    }
}

public class Tablica_osob {

    Tablica tablica = new Tablica();
    GUI1 gui1 = new GUI1();

    static public void main(String args[]) {
        Tablica_osob ap = new Tablica_osob();
        ap.tablica.Dodaj_osobe(ap.gui1.Wstaw());
        ap.tablica.Dodaj_osobe(ap.gui1.Wstaw());
        ap.gui1.Wyswietl(ap.tablica.Podaj_dane_tablicy());
    }
}
```



Nie mozna dodac wiecej danych do tablicy



4. Tablica obiektów – dodanie dziedziczenia, operacje wstawiania, wyświetlania i wyszukiwania

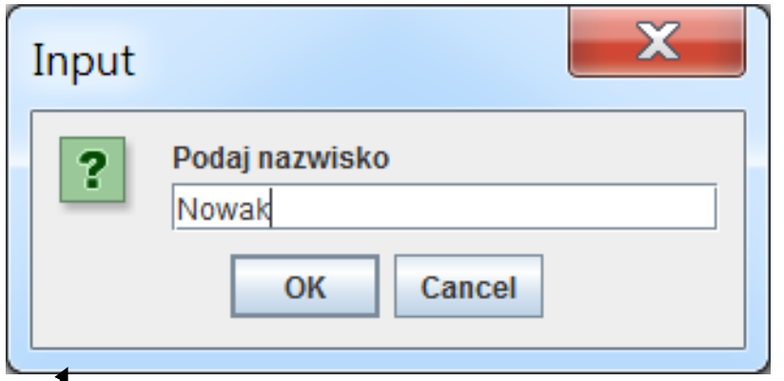
Przykład 4

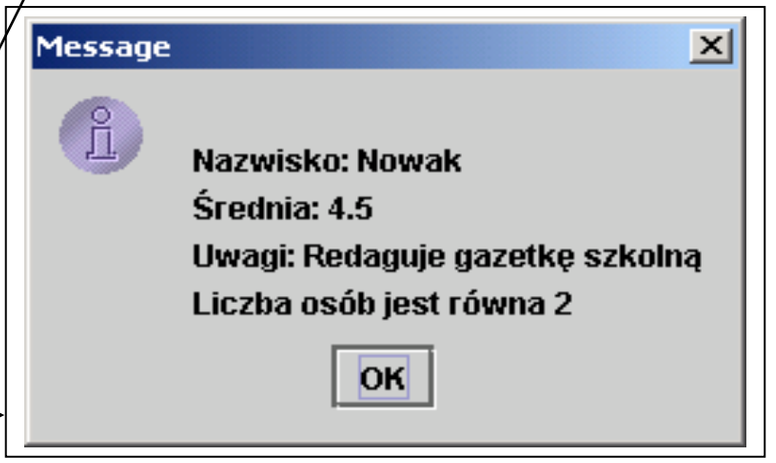
```
import javax.swing.JOptionPane;

class Osoba1
{//kod klasy Osoba1 z przykladu 2
}

class Osoba2 extends Osoba1{          // klasa Osoba2 posiada metodę sprawdzającą nazwisko

    public boolean Porownaj_nazwisko(String dana) {
        return nazwisko.equals(dana); }
}

class GUI1
 {//kod klasy GUI1 z przykladu 2
}

class GUI2 extends GUI1{              //klasa GUI2 posiada metode do wprowadzania nazwiska z klawiatury

   public String Podaj_nazwisko()
      {  return JOptionPane.showInputDialog(null, "Podaj nazwisko");   }
}
```

```
class Tablica {

    Osoba2 osoby[] = new Osoba2[N];              //modyfikacja typu elementów tablicy
    static int N = 2;
    int ile;

    public void Dodaj_osobe(String[] dane) {
        if (Pelna())
            return;
        Osoba2 nowa = new Osoba2();
        nowa.Nadaj_dane(dane);
        ile++;
        osoby[ile - 1] = nowa;
    }

    boolean Pelna() {
        return ile == N;
    }
    public String Podaj_dane_tablicy() {
        String dane_tablicy = "";
        for (Osoba2 osoba : osoby) {
            dane_tablicy += osoba.Podaj_dane() + "\n";
        }
       return dane_tablicy;
    }
}
```

```
class Tablica1 extends Tablica{        //klass Tablica1 posiada metode do wyszukania obiektu typu //Osoba2

    public String Szukaj(String nazwisko) {
      for (Osoba2 osoba : osoby)
          if (osoba.Porownaj_nazwisko(nazwisko))
              return osoba.Podaj_dane();
      return "Brak osoby";
    }
}

public class Tablica_osob_szukanie {

    Tablica1 tablica1 = new Tablica1();
    GUI2 gui2 = new GUI2();

    static public void main(String args[]) {
        Tablica_osob_szukanie ap = new Tablica_osob_szukanie();
        ap.tablica1.Dodaj_osobe(ap.gui2.Wstaw());
        ap.tablica1.Dodaj_osobe(ap.gui2.Wstaw());
        ap.gui2.Wyswietl(ap.tablica1.Podaj_dane_tablicy());
        String wynik=ap.tablica1.Szukaj(ap.gui2.Podaj_nazwisko());
        ap.gui2.Wyswietl(wynik);
    }
}
```

Dane wstawiono takie same jak w przykładzie 2





5. Tablica obiektów –wybór wstawiania, wyświetlania i wyszukiwania.

Wykorzystanie typu abstrakcyjnego Tablica w celu uniezależnienia klas korzystających z tej klasy od implementacji metod: Dodaj_osobe oraz Powieksz_tablice w klasach pochodnych.

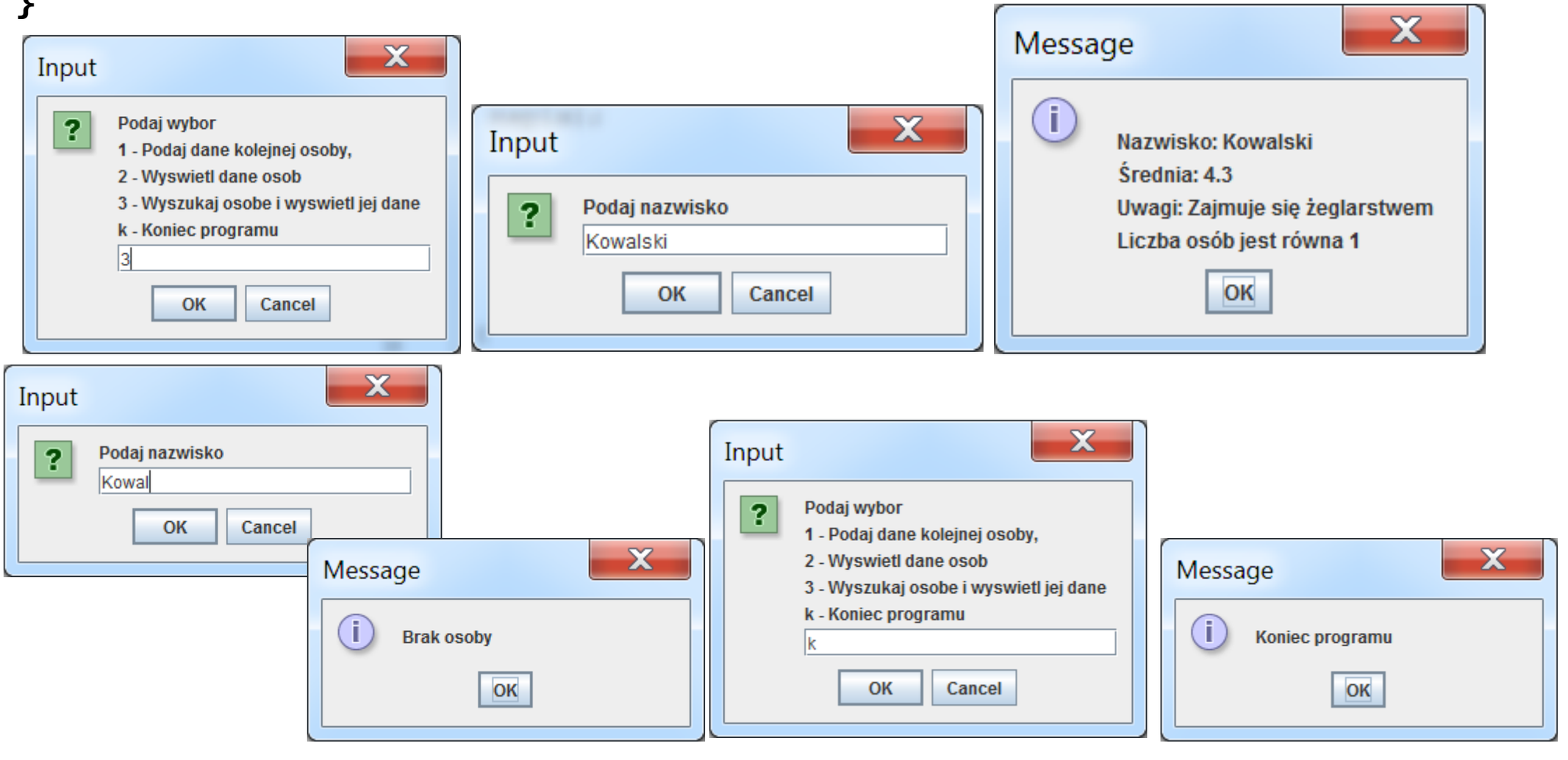
Przykład 5

```
class Osoba2
     {// kod klasy Osoba2 z przykładu 4
      }
 class GUI2
   {//kod klasy GUI2 z przykladu 4
   }

 abstract class Tablica
{   Osoba2 osoby[] = new Osoba2[N];
    static int N = 2;
    int ile;
    public abstract void Dodaj_osobe(String[] dane);
    public abstract void Powieksz_tablice();

    boolean Pelna() {       return ile == N;    }

    public String Podaj_dane_tablicy() {
        String dane_tablicy = "";
        for (int i=0;i<ile;i++)          //przy dowolnej teraz zawartosci tablicy nalezy odwołać sie
                                         //tylko do tylu elementów, ile wstawiono do tablicy
            dane_tablicy += osoby[i].Podaj_dane() + "\n";
        return dane_tablicy;  }
```

```
 public String Szukaj(String nazwisko) {
      for (int i=0;i<ile;i++)          //przy dowolnej teraz zawartosci tablicy nalezy odwolac sie
                                       //tylko do tylu elementów, ile wstawiono do tablicy
          if (osoby[i].Porownaj_nazwisko(nazwisko))
              return osoby[i].Podaj_dane();
      return "Brak osoby";
   }
}

class Tablica1 extends Tablica{

   @Override
   public void Dodaj_osobe(String[] dane) {
      if (Pelna())
          return;
      Osoba2 nowa = new Osoba2();
      nowa.Nadaj_dane(dane);
      ile++;
      osoby[ile - 1] = nowa;
   }

   @Override
    public void Powieksz_tablice()  {}
   }
```

```
class Tablica_osob_menu_ {//Klasa ta jest niezalezna od implementacji klasy Tablica
   Tablica tablica;
   GUI2 gui2;

   public Tablica_osob_menu_(Tablica tablica_, GUI2 gui_)
   {   tablica=tablica_;       gui2=gui_;    }

  public void menu()
   { String s;
     char ch;
     do {
        s = JOptionPane.showInputDialog(null, "Podaj wybor"
                + "\n1 - Podaj dane kolejnej osoby,"
                + "\n2 - Wyswietl dane osob"
                + "\n3 - Wyszukaj osobe i wyswietl jej dane"
                + "\nk - Koniec programu");
        ch = s.charAt(0);
       switch (ch) {
           case '1':   tablica.Dodaj_osobe(gui2.Wstaw());                          break;
           case '2':   gui2.Wyswietl(tablica.Podaj_dane_tablicy());                break;
           case '3':  String wynik = tablica.Szukaj(gui2.Podaj_nazwisko());
                      gui2.Wyswietl(wynik);                                        break;
           case 'k':   JOptionPane.showMessageDialog(null, "Koniec programu");  break;
           default:   JOptionPane.showMessageDialog(null, "Zla opcja");           break;
        }
     } while (ch != 'k');
     System.exit(0);    }}
```

```
public class Tablica_osob_menu
{
   public static void main(String args[])
   {
      Tablica_osob_menu_ tab=new Tablica_osob_menu_(new Tablica1(), new GUI2());
      tab.menu();
   }
}
```

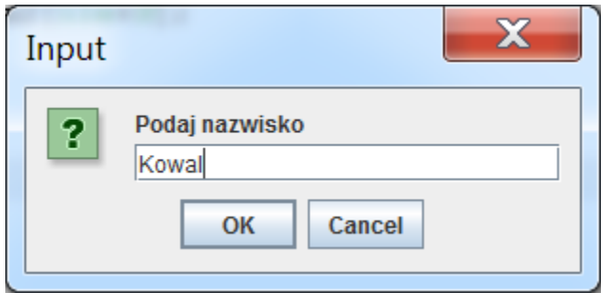


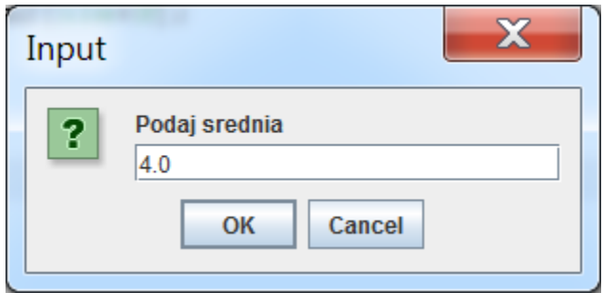
6. Tablica obiektów –wybór wstawiania, wyświetlania i wyszukiwania, zmiana rozmiarów tablicy – przykład wielużywalności kodu klasy Tablica_osob_menu_, niezależnej od implementacji klasy Tablica

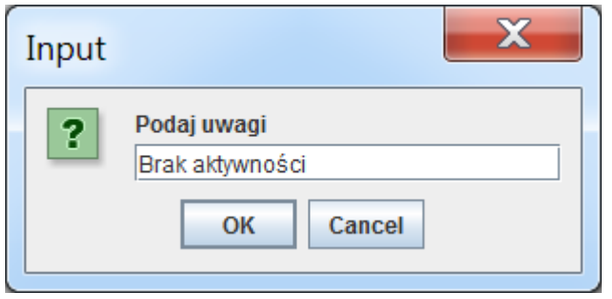
Przykład 6

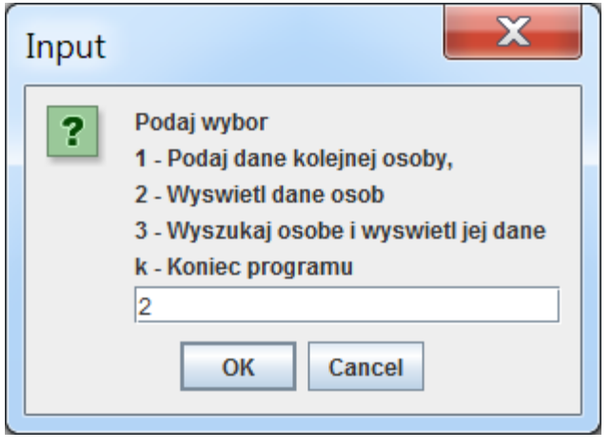
```
class Osoba2
    {// kod klasy Osoba2 z przykładu 4
    }

class GUI2
  {//kod klasy GUI2 z przykładu 4
  }

abstract class Tablica{
  //kod klasy Tablica z przykładu 5
}

  //wielużywalna klasa niezależna od kodu klas implementujących klasę Tablica
  class Tablica_osob_menu_
  {//kod klasy Tablica_osob_menu z przykładu 5
  }
```

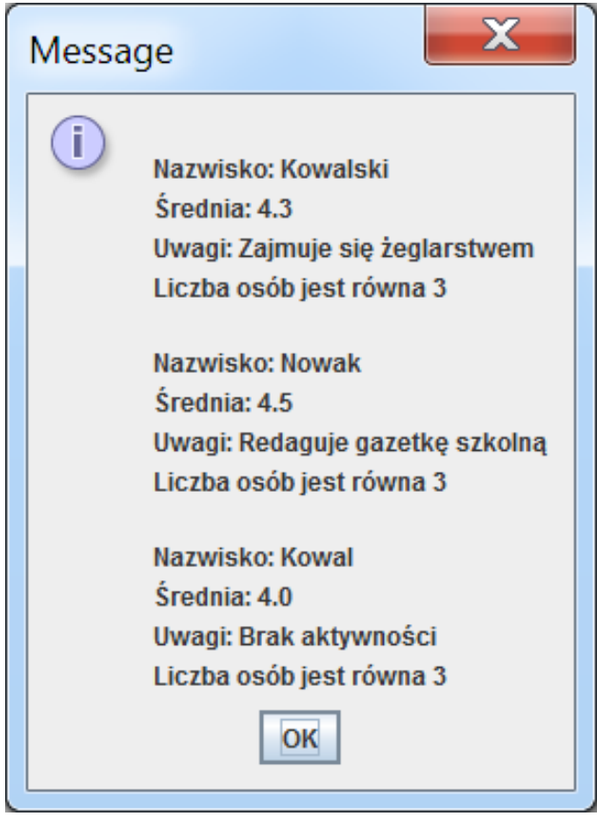
```
class Tablica2 extends Tablica{            // nowa implementacja klasy Tablica
   @Override
   public void Dodaj_osobe(String[] dane) {
      if (Pelna())
         Powieksz_tablice();
      Osoba2 nowa = new Osoba2();
      nowa.Nadaj_dane(dane);
      ile++;
      osoby[ile - 1] = nowa;
   }
@Override
   public void Powieksz_tablice()
   {  Osoba2 osobypom[]=new Osoba2[ile+N];   //tworzenie nowej lokalnej tablicy o rozmiarze ile +N
      int i=0;
      for (Osoba2 pom:osoby)                 //kopiowanie elementów z tablicy-atrybutu do nowej tablicy lokalnej
         osobypom[i++]=pom;
      osoby=osobypom;                        //tablica lokalna jest przypisana do tablicy – atrybutu obiektu,
   }                                         //zawartosc tablicy atrybutu jest  niedostepna i ulegnie usunieciu z pamieci
}                                            //a jej zawartosc jest zastapiona zawartoscia tablicy lokalnej

public class Tablica_osob_menu1
{
   public static void main(String args[])
   {  Tablica_osob_menu_ tab=new Tablica_osob_menu_(new Tablica2(), new GUI2());
      tab.menu();  }
}
```


Input
Podaj nazwisko
Kowal
OK
Cancel


Input
Podaj srednia
4.0
OK
Cancel


Input
Podaj uwagi
Brak aktywności
OK
Cancel


Input
Podaj wybor
1 - Podaj dane kolejnej osoby,
2 - Wyswietl dane osob
3 - Wyszukaj osobe i wyswietl jej dane
k - Koniec programu
2
OK
Cancel


Message
Nazwisko: Kowalski
Średnia: 4.3
Uwagi: Zajmuje się żeglarstwem
Liczba osób jest równa 3
Nazwisko: Nowak
Średnia: 4.5
Uwagi: Redaguje gazetkę szkolną
Liczba osób jest równa 3
Nazwisko: Kowal
Średnia: 4.0
Uwagi: Brak aktywności
Liczba osób jest równa 3
OK