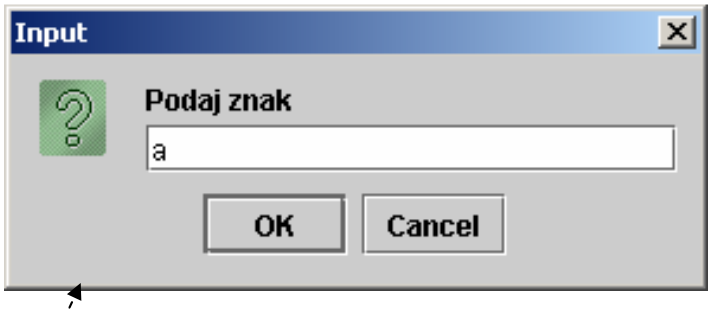
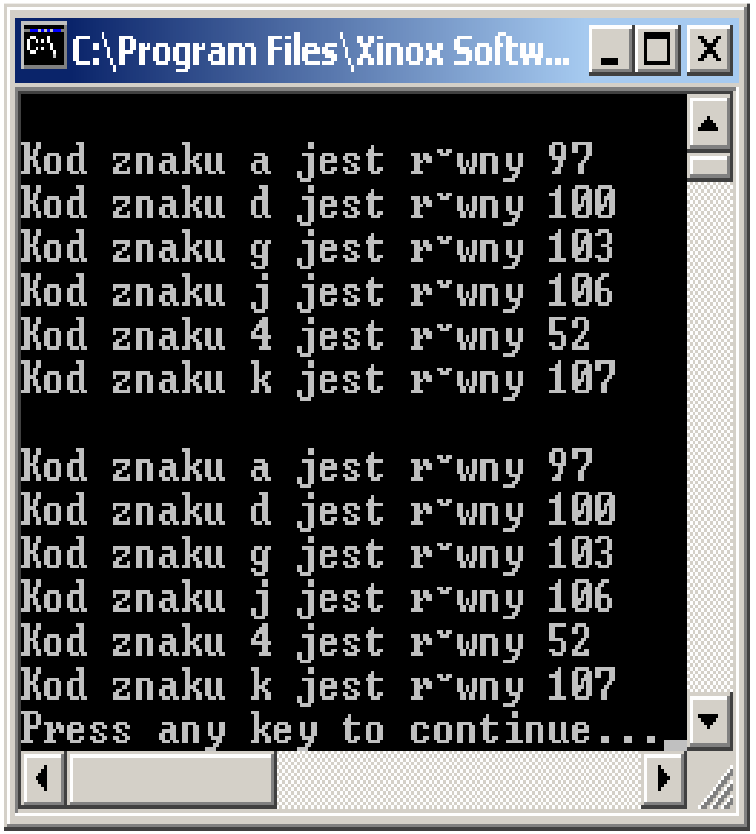
Przykład 1 – metody i atrybuty statyczne

```
import javax.swing.*;
import java.util.*;

public class Napis1
 { static String wynik;

   public static void Inicjuj()
       {wynik = "";}

   public static void Dopisz_do_wyniku(char ch)
      { wynik+="\nKod znaku "+ ch + " jest równy "+ (int)ch;  }

   public static void Rysuj_graficznie()
      { JOptionPane.showMessageDialog(null, wynik);   }

   public static void Rysuj_konsolowo()
     { System.out.println(wynik);  }

   public static void main(String[] args)
    { String s;
      char ch ='a';
      Inicjuj();
      while ( ch != 'k' )      //1 sposób podawania znaku
        { s=JOptionPane.showInputDialog(null, "Podaj znak");
          ch=s.charAt(0);
          Dopisz_do_wyniku(ch);
        }
      Rysuj_graficznie();
      Rysuj_konsolowo();
      System.out.println(wynik);
      System.exit(0);
 } }
```





```
import javax.swing.*;
import java.util.*;

public class Napis1
 {
  static String wynik;
   public static void Inicjuj()
       {wynik = "";}
   public static void Dopisz_do_wyniku(char ch)
      { wynik+="\nKod znaku "+ ch + " jest równy "+ (int)ch;  }
   public static void Rysuj_graficznie()
      { JOptionPane.showMessageDialog(null, wynik);   }
   public static void Rysuj_konsolowo()
     { System.out.println(wynik);  }
   public static void main(String[] args)
    { String s;
      char ch ='a';
      Inicjuj();
      while ( ch != 'k' )
        { s=JOptionPane.showInputDialog(null, "Podaj znak");
          ch=s.charAt(0);
          Dopisz_do_wyniku(ch);
        }
      Rysuj_graficznie();
      Rysuj_konsolowo();
      System.out.println(wynik);
      System.exit(0);
    }
 }
```

Nazwa klasy Napis1

Atrybut klasy typu **static** (składowa typu **static**) – istnieje przez cały czas działania programu w pamięci programu - **definicja atrybutu static**

Metody klasy Napis1 typu **static** (składowe funkcje klasy) – można je używać czyli wywoływać bez tworzenia obiektu klasy Napis1 – **definicje metod static**

Wywołania metod typu static klasy JOptionPane we metodzie main innych klas np. Napis1

Wywołania metod typu static klasy Napis1 we własnej metodzie main

Wywołania atrybutu typu static klasy Napis1.

nagłówek metody – funkcji składowej statycznej

typ **wynik bez typu** **nazwa** **lista parametrów ()** **ciało metody {}**

```
public static void Dopisz_do_wyniku(char ch)
{  wynik+="\nKod znaku "+ ch + " jest równy "+ (int)ch;
 }
```

Dostęp do **atrybutów statycznych i niestatycznych** jest **bezpośredni w metodach statycznych i niestatycznych własnej klasy**. Wyjątkiem jest metoda main dla składowych niestatycznych .

Uwaga: W metodzie typu static muszą być jedynie wywoływane atrybuty typu static i metody typu static

void – brak typu

```
Napis1.Rysuj_graficznie();
```

Wywołanie metody statycznej za pomocą nazwy klasy Napis1 w metodzie main. Jedynie w metodzie main dla metod typu **static** dodano do nazwy metody nazwę klasy Napis1 oraz operator wyboru **„.”. (Nie jest to obowiązkowe, czyli bez podania nazwy klasy metoda statyczna też może być wywołana w metodzie main własnej klasy)**

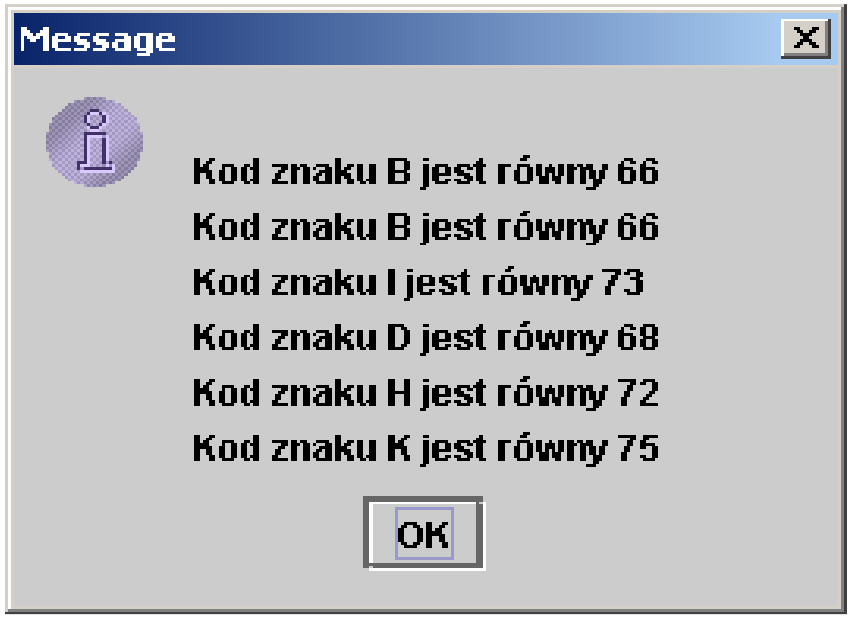
```
System.out.println(Napis1.wynik);
```

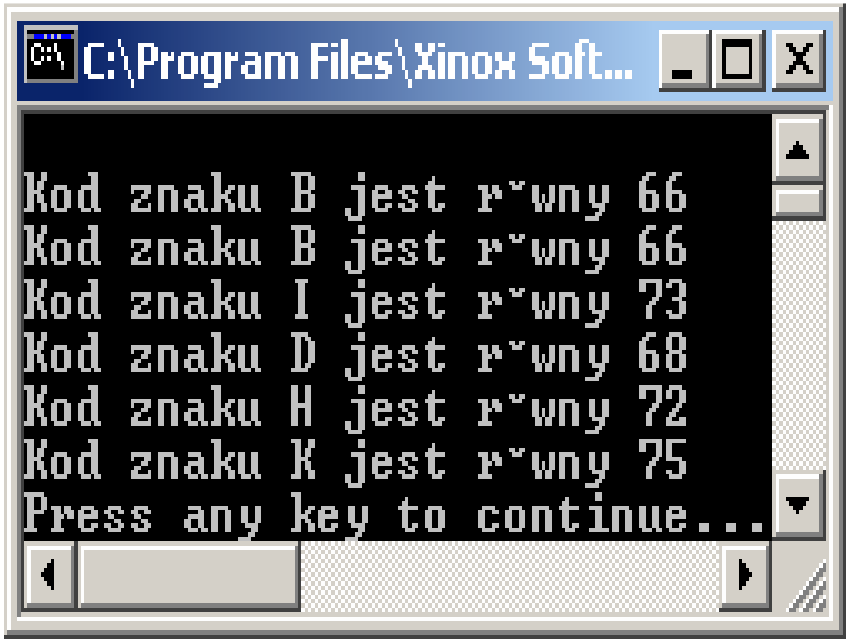
Wywołania atrybutu typu static klasy Napis1 w metodzie main. Jedynie w metodzie main dla atrybutów typu **static** dodano do nazwy metody nazwę klasy Napis1 oraz operator wyboru **„.”.(Nie jest to obowiązkowe, czyli bez podania nazwy klasy atrybut statyczny też może być wywołany w metodzie main własnej klasy)**

Obiektowy styl programowania - używanie metod w celu przetwarzania atrybutów obiektu - hermetyzacja

Przykład 2 – wielużywalność kodu klasy

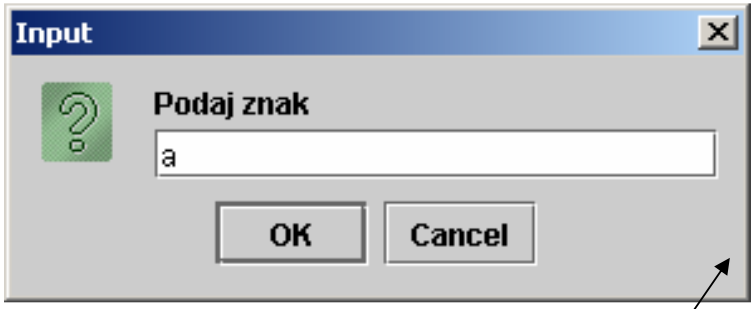
```
import javax.swing.*;
import java.util.*;
public class Napis2
 {
   static String wynik;
   public static void Inicjuj()
     {wynik = "";}
   public static void Dopisz_do_wyniku(char ch)
     {wynik+="\nKod znaku "+ ch + " jest równy "+ (int)ch;   }
  public static void Rysuj_graficznie()
     {JOptionPane.showMessageDialog(null, wynik);   }
  public static void Rysuj_konsolowo()
     { System.out.println(wynik);   }
 public static void main(String[] args)
   {
     char ch ='a';
     Inicjuj();
     while ( ch != 'K' )
      {       //2 sposób podawania znaku
       ch= (char)(Math.random()*11 +65);  znaku
       Dopisz_do_wyniku(ch);
      }
     Rysuj_graficznie();
     Rysuj_konsolowo();
     System.exit(0);
    }
 }
```

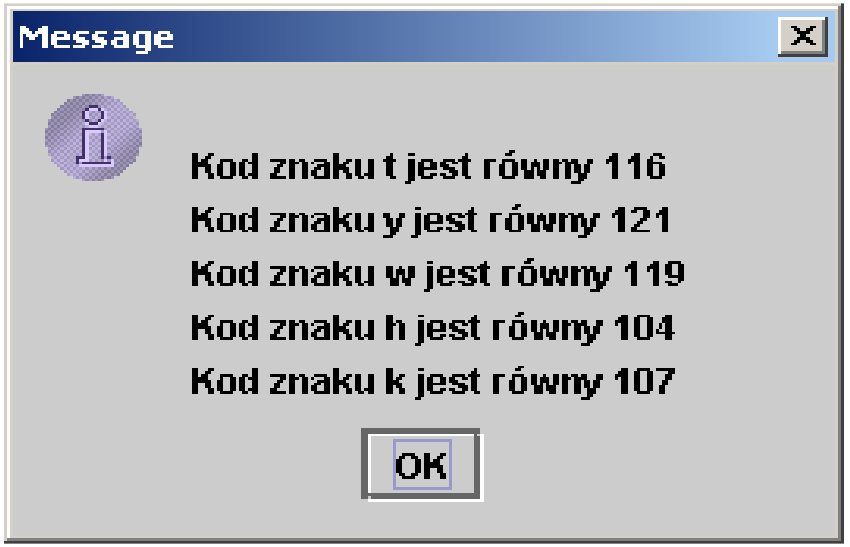


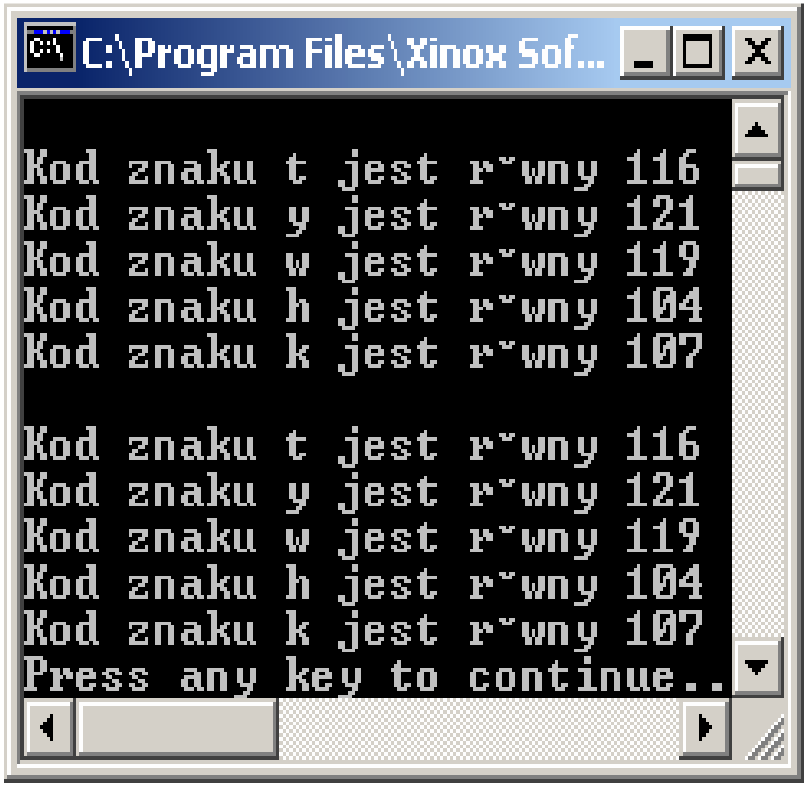


Przykład 3 – metody i atrybuty niestatyczne

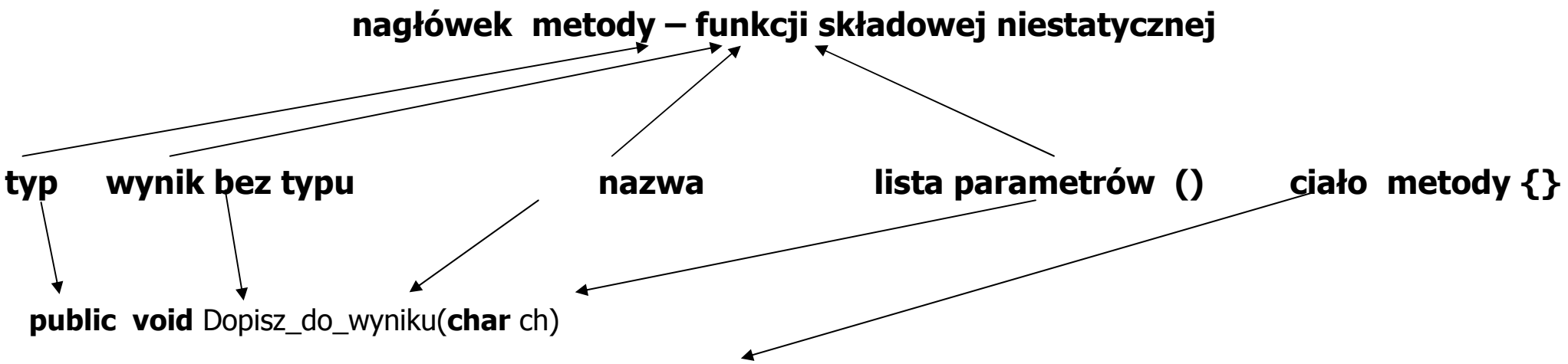
```
import javax.swing.*;
import java.util.*;

public class Napis3
 { String wynik;

   public void Inicjuj()
     { wynik = "";}

   public void Dopisz_do_wyniku(char ch)
     { wynik+="\nKod znaku "+ ch + " jest równy "+ (int)ch;}

   public void Rysuj_graficznie()
     {JOptionPane.showMessageDialog(null, wynik); }

   public void Rysuj_konsolowo()
     { System.out.println(wynik);}

   public static void main(String[] args)
    { String s;
      char ch ='a';
      Napis3 napis;
      napis = new Napis3();
      napis.Inicjuj();
      while ( ch != 'k' )
       { s =JOptionPane.showInputDialog(null, "Podaj znak");
         ch = s.charAt(0);
         napis.Dopisz_do_wyniku(ch);
       }
         napis.Rysuj_graficznie();
         napis.Rysuj_konsolowo();
         System.out.println(napis.wynik);
         System.exit(0);
    } }
```







Brak słowa **static** przy definicjach wymaga utworzenia obiektu typu Napis3.
Nazwa klasy Napis3 oznacza typ obiektu.
Obiekt powstaje po wywołaniu operatora **new**



Napis3 napis;

Referencja typu Napis3 niezainicjowana – brak obiektu typu Napis3

napis = **new** Napis3();

Referencja typu Napis3 zainicjowana – utworzono obiekt typu Napis3 za pomocą operatora **new**

napis.Rysuj_graficznie();

Wywołanie metody **niestatycznej** Rysuj_graficznie() za pomocą referencji **napis** do obiektu typu Napis3 oraz operatora wyboru „**.**”**-obowiązkowe w metodzie main własnej klasy dla metod niestatycznych**

System.out.println(napis.wynik);

Wywołania atrybutu typu niestatycznego wynik klasy Napis1 w metodzie main za pomocą referencji **napis** do obiektu typu Napis3 oraz operatora wyboru „**.**” **– obowiązkowe w metodzie main własnej klasy dla atrybutów niestatycznych**

Przykład 4 – metody i atrybuty statyczne oraz niestatyczne

```
import javax.swing.*;
import java.util.*;

public class Napis4
 {
  String wynik = "";
  static int ile_obiektow = 0;

  public void Inicjuj()
    { ile_obiektow++;
      wynik+="";
    }

  public  void Dopisz_do_wyniku(String lan)
    {  wynik+=lan+"\n"; }

  public  void Rysuj_graficznie()
    { JOptionPane.showMessageDialog(null, wynik); }

  public  void Rysuj_konsolowo()
    { System.out.println(wynik); }
```

1) Metoda niestatyczna, oprócz atrybutów niestatycznych i metod niestatycznych, może używać atrybutów statycznych i metod statycznych danej klasy.
2) Należy utworzyć obiekt, aby wywołać metodę niestatyczną klasy tego obiektu.
3) Atrybut statyczny należy do klasy i dlatego jest wspólny dla wszystkich obiektów. Dlatego może być on używany w dowolnym kontekście: statycznym i niestatycznym
4) W metodzie statycznej można używać tylko atrybutów statycznych i metod statycznych danej klasy

1) Każdy obiekt, który używa w metodzie statycznej lub niestatycznej atrybutu statycznego, używa tego atrybutu jako wspólnego:
2) Każdy obiekt ma własny atrybut **wynik** oraz wspólny atrybut

```
public static void main(String[] args)
   { String s;
     Napis4 napis1, napis2;
     napis1=new Napis4();
     napis2=new Napis4();
     napis1.Inicjuj();
     napis2.Inicjuj();
     s=JOptionPane.showInputDialog(null,"Podaj lancuch do obiektu 1");
     napis1.Dopisz_do_wyniku(s);
     s=JOptionPane.showInputDialog(null,"Podaj lancuch do obiektu 2");
     napis2.Dopisz_do_wyniku(s);


     napis1.Rysuj_graficznie();


     napis2.Rysuj_graficznie();

     napis1.Rysuj_konsolowo();
     napis2.Rysuj_konsolowo();
     System.out.println("Sa "+napis1.ile_obiektow+" obiekty");
     System.out.println("Sa "+napis2.ile_obiektow+" obiekty");
     System.out.println("Sa "+Napis4.ile_obiektow+" obiekty");
     System.exit(0);
   }// nazwę obiektów napis1 I napis2 można pominąć przy odwołaniu do atrybutu
}// ile_obiektow, ponieważ jest to statyczny atrybut (wspólny dla obu obiektów)
```