Ćw.1. Czcionki systemowe

a) Utworzyć aplet, który wyświetli listę dostępnych czcionek systemowych. Klasa **Toolkit** (z pakietu **java.awt**) dostarcza informacji o systemie. Metoda **getFontList** z **Toolkit** określa dostępne czcionki systemowe. W klasie **Graphics** (pakiet **java.awt**) metodą **setFont** wybieramy określoną czcionkę dla wcześniej zdefiniowanego obiektu **Font** a metodą **setColor** ustalamy jej kolor.

```
import java.awt.*;import java.applet.*;
public class fonty extends Applet
{  public void paint(Graphics g)
  {      Toolkit  narzedzia=Toolkit.getDefaultToolkit();   // pobieramy zestaw narzędzi systemowych
         String tablica_fontow[]=narzedzia.getFontList(); // pobieramy do tablicy listę czcionek systemowych
         int ile=tablica_fontow.length;                    // ustalamy rozmiar tablicy fontów
         Font f=new Font(tablica_fontow[0], Font.BOLD, 20);
                  //kolejne parametry funkcji Font: nazwa czcionki (pobrana z tablicy fontów), styl, rozmiar
                  //styl: Font.PLAIN, Font.BOLD, Font.ITALIC, oraz połączenia np.Font.BOLD+Font.ITALIC
                  //styl: można również określić za pomocą wartości 0,1,2,3
         g.setFont(f);  //ustalenie fontu dla obiektu g
         g.setColor(Color.blue);       //ustalenie koloru pierwszoplanowego dla obiektu g
         g.drawString(tablica_fontow[0],20, 25);   //wyświetlenie napisu - nazwy czcionki umieszczonej jako
                                                    //element o indeksie 0 w tablicy fontów
   }
}
```

Wynik działania powyższego apletu przedstawia rys.1

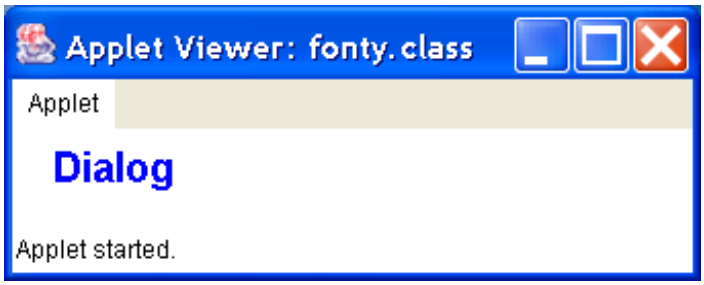
b) Zmodyfikować aplet tak, aby wyświetlał wszystkie czcionki dostępne w systemie (czyli wszystkie elementy tablicy **tablica_fontów**) - zastosować pętlę for:

```
for(int i=0;i<ile;i++)
{   //instrukcje do wykonania w pętli for
 }
```

Przykładowy efekt działania przedstawia rys.2

c) Zmodyfikować kod programu tak, aby kolejne napisy były wyświetlane za pomocą różnych dostępnych styli (skorzystać z możliwości definiowania stylu za pomocą wartości 0, 1, 2, 3 i w pętli for zastosować dzielenie modulo: **i%4**) i o rozmiarach czcionek kolejno 20, 25, 30, ...(rys.3).

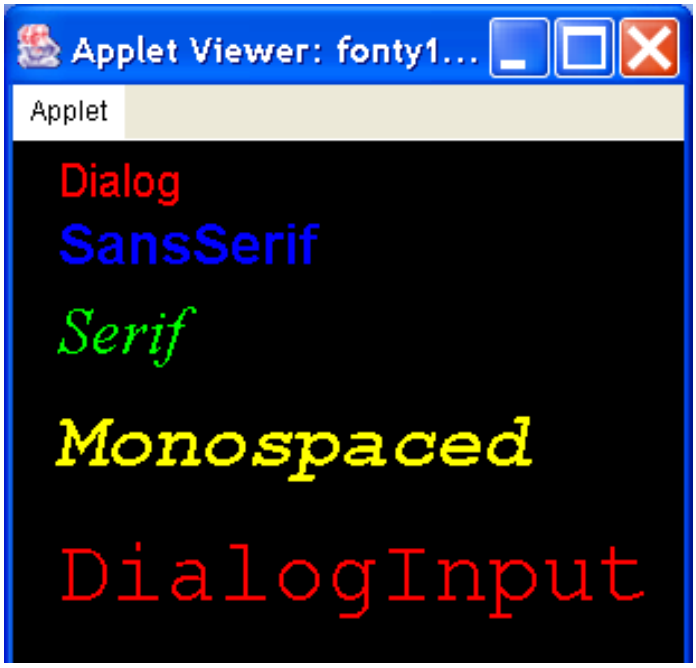
d) zmienić tło apletu na czarne, zdefiniować tablicę zawierającą kolory np. czerwony, zielony, niebieski, żółty:
Color[] kol={Color.red,Color.blue,Color.green, Color.yellow};
oraz zmodyfikować działanie klasy tak, aby kolejne napisy pojawiały się w kolorach pobieranych z utworzonej tablicy kolorów (rys.4).



Rys.1.

Rys.2

Rys.3.



Rys.4

Ćw.2. Animacja tekstu

Napisać aplet **animacja**, która spowoduje wyświetlanie napisu **Witaj!!!**:

- przesuwającego się od lewej do prawej krawędzi okna,
- spadającego w dół począwszy od górnej krawędzi okna,
- poruszającego się po przekątnej okna

Do ustalenia wymiarów okna apletu wykorzystać własności: **getWidth(); getHeight();**

Wykorzystać:

a) **pętlę animacji**, w której należy zmieniać położenie wyświetlanego tekstu, wyświetlać ten tekst w kolorze pierwszoplanowym a następnie po upływie czasu określonego metodą **sleep** wyświetlać napis w kolorze tła

b) metodę **repaint();** która wywołuje ponownie metodę **paint** ze zmodyfikowanym położeniem tekstu

W celu zatrzymania programu na określoną liczbę milisekund skorzystać z metody **sleep** wołanej na rzecz wątku (programu) **Thread**. Ponieważ metoda **sleep** może generować błąd-wyjątek, należy tę metodę wywołać za pomocą konstrukcji **try-catch** w postaci:

```
try  {
        Thread.sleep(250);//zatrzymanie programu na 250 ms
    }  catch (InterruptedException e) {}  //obsługa ewentualnego błędu
```