Ocena tętna spoczynkowego

Problem

Monitorowanie stanu zdrowia jest ważne, jednym z jego elementów jest analiza pomiarów tętna. Tętno inaczej nazywane jest też pulsem. Przedstawione poniżej informacje na temat prawidłowej wartości tętna mają *charakter poglądowy i mogą nie mieć pełnego związku z wiedzą medyczną*, stanowią one jedynie specyfikację potrzebną do realizacji ćwiczenia programistycznego.

Tętno spoczynkowe powinno mieć wartość z zakresu 60-100 uderzeń na minutę (bpm). Wartość poniżej 60 uważana jest za tętno zbyt niskie, chyba że dana osoba uprawia sport i jest wytrenowana. Wartość poniżej 40 bpm jest niepokojąca, wymaga konsultacji lekarskiej, podobnie jak wartość większa niż 100 bpm. Wartość tętna spoczynkowego większa niż 160 wymaga natychmiastowej kontroli lekarskiej. Wartość tętna jest liczbą całkowitą.

Ćwiczenie

Proszę napisać program, który prosi użytkownika o wprowadzenie wartości tętna spoczynkowego a następnie informuje użytkownika programu jaka jest interpretacja wprowadzonej wartości, używając zakresów i komunikatów opisanych w Tabeli 1.

Tabela 1 Zakresy wartości tętna i komunikaty programu

	Zakres wartości tętna [bpm]	Komunikat
1	mniej niż 40	Tętno zbyt niskie, skontaktuj się z lekarzem
2	od 40 do 59	Tętno niskie, ale jeżeli trenujesz to jest OK
3	od 60 do 99	Tętno w normie
4	od 100 do 159	Tętno zbyt wysokie, skontaktuj się z lekarzem
5	powyżej 160	Konieczna natychmiastowa konsultacja lekarska

Program ma właściwie zareagować na wartości skrajne i nieprawidłowe (Tabela 2).

Tabela 2 Wartości spoza zakresu oraz wartości nieprawidłowe

	Wprowadzona wartość	Komunikat

1	Nie jest liczbą całkowitą	Wartość tętna powinna być liczbą całkowitą
2	Jest liczbą ujemną	Wartość tętna powinna być większa od zera
3	Jest równa zero	Zerowa wartość tętna? To oznacza zatrzymanie akcji serca, co prowadzi do śmierci
4	Jest większa od 220	Tak wysoka wartość tętna nie jest prawidłowa

Testy

Proszę przetestować program wprowadzając wybrane wartości tętna dla każdego z zakresów opisanych w Tabeli 1, uwzględniając wartości graniczne oraz wprowadzając wartości opisane w Tabeli 2. Proszę sprawdzić czy działanie programu jest zgodne z zapisami zawartymi w tabelach 1 i 2.

Przykładowe rozwiązania – wersja pierwsza

Przykładowe rozwiązania nie obejmują wszystkich przypadków opisanych w specyfikacji zadania.

Język Python

Wersja 1 - wykorzystanie *if-else*

```
print("Analizuję wartość tętna spoczynkowego")
puls = int(input("Tętno spoczynkowe: "))
if puls < 60:
    print("Tętno niskie, ale jeżeli trenujesz to jest OK")
else:
    if puls < 100:
        print("Tętno w normie")
    else:
        if puls < 140:
            print("Tętno jest trochę za wysokie")
        else:
            print("Tętno zbyt wysokie, skontaktuj się z lekarzem")
```

Wersja 2 - wykorzystanie *if-elif-else*

```
print("Analizuję wartość tętna spoczynkowego")
puls = int(input("Tętno spoczynkowe: "))
if puls < 60:
    print("Tętno niskie, ale jeżeli trenujesz to jest OK")
elif puls < 100:
    print("Tętno w normie")
elif puls < 140:
```

```
    print("Tętno jest trochę za wysokie")
else:
    print("Tętno zbyt wysokie, skontaktuj się z lekarzem")
```

Wersja 3 - wykorzystanie instrukcji *try-except*

```
print("Analizuję wartość tętna spoczynkowego")
try:
    puls = int(input("Tętno spoczynkowe: "))
    if puls < 60:
        print("Tętno niskie, ale jeżeli trenujesz to jest OK")
    elif puls < 100:
        print("Tętno w normie")
    elif puls < 140:
        print("Tętno jest trochę za wysokie")
    else:
        print("Tętno zbyt wysokie, skontaktuj się z lekarzem")
except:
    print("Wartość tętna powinna być liczbą całkowitą")
```

Język Java

Wersja 1 - wykorzystanie instrukcji *if-else*

```
import java.util.Scanner;

public class Main
{
    public static void main(String[] args)
    {
        int puls;
        Scanner wejscie = new Scanner(System.in);
        System.out.println("Analizuję pomiar tętna spoczynkowego");
        System.out.print("Podaj wartość tętna: ");
        puls = wejscie.nextInt();
        if(puls < 60)
            System.out.println("Niskie tętno");
        else
            if(puls < 100)
                System.out.println("Tętno w normie");
            else
                if(puls < 140)
                    System.out.println("Tętno za wysokie");
                else
                    System.out.println("Tętno mocno zbyt wysokie");
    }
}
```

Wersja 2 - wykorzystanie instrukcji *try-catch*

W przypadku wprowadzenia przez użytkownika napisu nie będącego prawidłową liczbą całkowitą, wygenerowany zostanie wyjątek przez metodę *nextInt*. Instrukcja *try-catch* pozwala na obsługę tego wyjątku.

```
import java.util.Scanner;

public class Main
{
    public static void main(String[] args)
    {
        int puls;
        Scanner wejscie = new Scanner(System.in);
        try
        {
            System.out.println("Analizuję pomiar tętna spoczynkowego");
            System.out.print("Podaj wartość tętna: ");
            puls = wejscie.nextInt();
            if(puls < 60)
                System.out.println("Niskie tętno");
            else
                if(puls < 100)
                    System.out.println("Tętno w normie");
                else
                    if(puls < 140)
                        System.out.println("Tętno za wysokie");
                    else
                        System.out.println("Tętno mocno zbyt wysokie");
        }
        catch(Exception e)
        {
            System.out.println("Nieprawidłowa liczba");
        }
    }
}
```

Wersja 3 - wykorzystanie metody *hasNextInt* klasy Scanner

Przypadek wprowadzenia przez użytkownika napisu nie będącego prawidłową liczbą całkowitą można wykryć wykorzystując metodę *hasNextInt* klasy *Scanner*.

```
import java.util.Scanner;

public class Main
{
    public static void main(String[] args)
    {
        int puls;
```

```
        Scanner wejscie = new Scanner(System.in);
        System.out.println("Analizuję pomiar tętna spoczynkowego");
        System.out.print("Podaj wartość tętna: ");
        if(!wejscie.hasNextInt())
            System.out.println("Nieprawidłowa liczba");
        else
        {
            puls = wejscie.nextInt();
            if(puls < 60)
                System.out.println("Niskie tętno");
            else
                if(puls < 100)
                    System.out.println("Tętno w normie");
                else
                    if (puls < 180)
                        System.out.println("Tętno za wysokie");
                    else
                        System.out.println("Tętno mocno zbyt wysokie");
        }
    }
}
```

Wersja 4 -formatowanie if-else w stylu elif

Język Java nie zawiera aktualnie instrukcji warunkowych pozwalających na stosowanie frazy *elif*. Często zapisuje się jednak sekwencje zagnieżdżonych instrukcji warunkowych w sposób podobny do wersji wykorzystujących *elif*.

Zamiast:

```
if(puls < 60)
   System.out.println("Niskie tętno");
else
   if(puls < 100)
      System.out.println("Tętno w normie");
   else
      if(puls < 180)
         System.out.println("Tętno za wysokie");
      else
         System.out.println("Tętno mocno zbyt wysokie");
```

piszemy:

```
if (puls < 60)
    System.out.println("Niskie tętno");
else if (puls < 100)
    System.out.println("Tętno w normie");
else if (puls < 180)
    System.out.println("Tętno za wysokie");
else
```

```
    System.out.println("Tętno mocno zbyt wysokie");
```

Ta wersja jest czytelniejsza, sprzyja temu stosowanie spacji pomiędzy słowem kluczowym *if* a nawiasem otwierającym.

Rozszerzenie

Proszę rozszerzyć program o możliwość ponownego wprowadzenia niepoprawnej wartości tętna – nie będącej liczbą całkowitą, wartości ujemnej, zerowej lub przekraczającej wartość 220 (zgodnie z tabelą 2). W przypadku wykrycia takiej wartości program powinien zapytać ponownie o nową wartość, ignorując wprowadzoną uprzednio wartość nieprawidłową. Powinno to być powtarzane tak długo, aż użytkownik wprowadzi wartość prawidłową.

Przykładowe rozwiązania – wersja rozszerzona

Możliwość powtórnego wprowadzenia niepoprawnej wartości wymaga wprowadzenia iteracji, pozwalającej na wielokrotne powtórzenie sekwencji *wprowadź i skontroluj prawidłowość*. Przykładowe rozwiązania nie obejmują wszystkich przypadków opisanych w specyfikacji zadania.

Język Python

W przypadku języka Python trzeba wykorzystać iterację while, używając warunku zawsze prawdziwego otrzymujemy pętlę, z której wyskakujemy instrukcję *break*. Wyskok następuje wtedy, gdy żaden z testów sytuacji nieprawidłowych się nie powiedzie. Po opuszczeniu iteracji while mamy pewność, że wprowadzona wartość jest prawidłowa (nie spełnia kryteriów określonych w tabeli 2) i można dokonać oceny na podstawie przedziałów z tabeli 1.

```
print("Analizuję wartość tętna spoczynkowego")
while True:
    try:
        puls = int(input("Tętno spoczynkowe: "))
        if puls < 0:
            print("Wartość tętna powinna być większa od zera")
        elif puls == 0:
            print("Zerowa wartość tętna? To oznacza zatrzymanie akcji serca")
        elif puls > 220:
            print("Tak wysoka wartość tętna nie jest prawidłowa")
        else:
            break
    except:
        print("Wartość tętna powinna być liczbą całkowitą")
```

```
if puls < 60:
    print("Niskie tętno")
elif puls < 100:
    print("Tętno w normie")
elif puls < 180:
    print("Tętno za wysokie")
else:
    print("Tętno mocno zbyt wysokie")
```

Język Java

W przypadku języka można wykorzystać rozwiązanie analogiczne do przedstawionego w przykładzie zapisanym w języku Python. Jednak język Java pozwala na wykorzystanie instrukcji iteracyjnej *do-while*, która pozwala na zbudowanie konstrukcji bez pętli i bez wykorzystania instrukcji *break*. Wyjście z iteracji *do-while* nadzoruje zmienna boolowska *poprawnaWartosc*, która otrzymuje startową wartość *false*, co oznacza domniemanie błędnych danych wpisanych przez użytkownika. Sekwencja instrukcji *if-else* pozwala na ocenę sytuacji błędnych, gdy wykonanie programu trafi do ostatniej frazy *else*, wprowadzona wartość jest prawidłowa, zmiennej poprawnaWartosc nadawana jest wartość true, co pozwala na opuszczenie iteracji *do-while*.

W przypadku gdy funkcja *hasNextInt* klasy *Scanner* zakończy się z rezultatem *false*, wprowadzony przez użytkownika łańcuch znaków nie zawiera liczby i musi zostać odczytany z bufora klawiatury, służy do tego funkcja *next* klasy *Scanner*. Warto zwrócić uwagę na to, że ten odczyt to po prostu wyczyszczenie nieprawidłowych danych wejściowych, które nie są przypisywane do żadnej zmiennej. W przypadku gdy funkcja *hasNextInt* zakończy się z rezultatem *true*, odczyt prawidłowej liczby zostanie zrealizowany z wykorzystaniem funkcji *nextInt*, a odczytana wartość zostanie przypisana do zmiennej *puls*.

Uwaga: zbyt długie komunikaty zostały skrócone (znaki ...) aby nie zaciemniać struktury kodu przełamanymi liniami.

```
import java.util.Scanner;

public class Main
{
    public static void main(String[] args)
    {
        int puls = 0;
        boolean poprawnaWartosc = false;

        Scanner wejscie = new Scanner(System.in);
```

```
        System.out.println("Analizuję pomiar tętna spoczynkowego");
        do
        {
            System.out.print("Podaj wartość tętna: ");
            if (!wejscie.hasNextInt())
            {
                System.out.println("Nieprawidłowa liczba");
                wejscie.next();
            }
            else
            {
                puls = wejscie.nextInt();
                if (puls < 0)
                    System.out.println("Wartość tętna powinna być ...");
                else if (puls == 0)
                    System.out.println("Zerowa wartość tętna?");
                else if (puls > 220)
                    System.out.println("Tak wysoka wartość tętna ...");
                else
                    poprawnaWartosc = true;
            }
        }
        while (!poprawnaWartosc);

        if (puls < 60)
            System.out.println("Niskie tętno");
        else if (puls < 100)
            System.out.println("Tętno w normie");
        else if (puls < 180)
            System.out.println("Tętno za wysokie");
        else
            System.out.println("Tętno mocno zbyt wysokie");

    }
}
```