

SPRAWOZDANIE NR 1			
Nazwa ćwiczenia	#1 PODSTAWY PROGRAMOWANIA		 POLITECHNIKA BYDGOSKA Wydział Telekomunikacji, Informatyki i Elektrotechniki
Przedmiot	PROGRAMOWANIE W ŚRODOWISKU WINDOWS		
Student grupa	Sara Górską gr. 1		
Data ćwiczeń	18.03.2023	18.03.2023	Data oddania sprawozdania
Ocena, uwagi			

Link do github, na którym znajdują się kody programów:

<https://github.com/sargor000/Programowanie-w-srodowisku-Windows/tree/Laboratorium-%231>

Opis dotyczący wszystkich zadań

Programy zostały napisane w języku Java i jest wykonywany w środowisku IntelliJ IDEA w systemie Windows. IntelliJ IDEA to popularne i wydajne środowisko programistyczne do tworzenia aplikacji Java. System Windows jest jednym z najpopularniejszych systemów operacyjnych na świecie.

Dzięki zastosowaniu języka Java, program ten jest przenośny i może być uruchamiany na wielu różnych platformach, w tym na systemach Windows, macOS, Linux i innych. Java jest również popularnym językiem programowania w branży informatycznej.

Środowisko IntelliJ IDEA jest jednym z najlepszych i najpopularniejszych środowisk programistycznych dla języka Java, oferującym wiele narzędzi i funkcji ułatwiających proces tworzenia aplikacji. W połączeniu z systemem Windows, daje to bardzo wydajne i efektywne narzędzie dla programistów Java.

Zadania do wykonania

Ćwiczenie 1

Napisz program, w którym użytkownik zgaduje losową liczbę całkowitą X. Po każdej próbie program powinien wyświetlić komunikat, czy podana liczba jest większa, mniejsza bądź równa szukanej X.

Kod programu

Klasa Main

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

        System.out.println("Hello world!");
        int nowaLiczbaDoSprawdzania;
        int podajLiczbedoSprawdzenia = 0;

        LosowanieLiczby nowaLiczba = new LosowanieLiczby();
        nowaLiczbaDoSprawdzania = nowaLiczba.wylosujLiczbe();

        do {
            LosowanieLiczby podajLiczbe = new LosowanieLiczby();
            podajLiczbedoSprawdzenia = podajLiczbe.pobierzLiczbe();

            if (nowaLiczbaDoSprawdzania > podajLiczbedoSprawdzenia) {
                System.out.println("Twoja liczba jest za mała.");
            }
        }
```

```

        else if (nowaLiczbedoSprawdzania < podajLiczbedoSprawdzenia){
            System.out.println("Twoja liczba jest za duża.");
        }
        else {
            System.out.println("Zgadłeś liczbę!");
        }
    }
    while (nowaLiczbedoSprawdzania != podajLiczbedoSprawdzenia);
}
}

```

Klasa LosowanieLiczby

```

import java.util.Random;
import java.util.Scanner;
public class LosowanieLiczby {
    public int wylosujLiczbe()
    {
        Random generator = new Random();
        int wylosowanaLiczba = generator.nextInt(10) + 1;
        //System.out.println("Wylosowana liczba: " + wylosowanaLiczba);
        return wylosowanaLiczba;
    }
    public int pobierzLiczbe()
    {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Podaj liczbę całkowitą: ");
        int liczba = scanner.nextInt();
        System.out.println("Podana liczba to: " + liczba);
        return liczba;
    }
}

```

Wynik kodu programu

Przykład 1

```

"C:\Program Files\Java\jdk-17.0.5\bin\java.exe"
Hello world!
Podaj liczbę całkowitą: 6
Podana liczba to: 6
Twoja liczba jest za duża.
Podaj liczbę całkowitą: 3
Podana liczba to: 3
Twoja liczba jest za mała.
Podaj liczbę całkowitą: 4
Podana liczba to: 4
Zgadłeś liczbę!

```

Przykład 2

```

"C:\Program Files\Java\jdk-17.0.5\bin\java.exe"
Hello world!
Podaj liczbę całkowitą: 9
Podana liczba to: 9
Twoja liczba jest za mała.
Podaj liczbę całkowitą: 10
Podana liczba to: 10
Zgadłeś liczbę!

```

Opis działania programu i wykorzystanych technik programistycznych

Na początku program losuje liczbę, którą użytkownik będzie musiał odgadnąć. Następnie, za pomocą pętli do-while, program pyta użytkownika o wprowadzenie liczby i porównuje ją z wylosowaną liczbą.

Jeśli użytkownik poda liczbę mniejszą niż wylosowana, program wypisze "Twoja liczba jest za mała." i poprosi o podanie kolejnej liczby. W przypadku, gdy podana liczba jest większa od wylosowanej, program wypisze "Twoja liczba jest za duża." i poprosi o podanie kolejnej liczby. W przypadku, gdy użytkownik odgadnie liczbę, program wypisze "Zgadłeś liczbę!" i zakończy działanie.

Klasę LosowanieLiczby, składa się z dwóch metod: wylosujLiczbe() i pobierzLiczbe().

Metoda wylosujLiczbe() generuje losową liczbę całkowitą z przedziału od 1 do 10 przy pomocy klasy Random z pakietu java.util. Następnie zwraca tę liczbę. Metoda pobierzLiczbe() prosi użytkownika o podanie liczby całkowitej przy pomocy klasy Scanner. Następnie wyświetla jest ta liczba. Liczba jest zwracana.

Ćwiczenie 2

Napisz program zliczający liczbę unikatowych wartości całkowitych wprowadzonych przez użytkownika.

Kod programu

Klasa Main

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello world!");
        LiczenieUnikatowychLiczb liczba = new LiczenieUnikatowychLiczb();
    }
}
```

Klasa LiczenieUnikatowychLiczb

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

public class LiczenieUnikatowychLiczb {
    {
        Scanner scanner = new Scanner(System.in);
        Set<Integer> liczby = new HashSet<>();

        System.out.println("Podaj liczby całkowite (wprowadź 0, aby zakończyć):");

        int liczba = scanner.nextInt();
        while (liczba != 0) {
            liczby.add(liczba);
            liczba = scanner.nextInt();
        }
        System.out.println("Liczba unikatowych wartości: " + liczby.size());
    }
}
```

Wynik kodu programu

Przykład 1

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe" "-javaag
```

```
Hello world!
```

```
Podaj liczby całkowite (wprowadź 0, aby zakończyć):
```

```
1
```

```
2
```

```
3
```

```
0
```

```
Liczba unikatowych wartości: 3
```

Przykład 2

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe" "-ja
```

```
Hello world!
```

```
Podaj liczby całkowite (wprowadź 0, aby zakończyć):
```

```
11
```

```
1
```

```
2
```

```
2
```

```
3
```

```
4
```

```
0
```

```
Liczba unikatowych wartości: 5
```

Opis działania programu i wykorzystanych technik programistycznych

W klasie `LiczenieUnikatowychLiczb` importuje klasę `HashSet`, która umożliwia przechowywanie unikatowych wartości w zbiorze. `Java.util.Scanner` to klasa `Scanner`, która umożliwia pobieranie danych wprowadzanych przez użytkownika z konsoli. `Java.util.Set` to interfejs `Set`, który reprezentuje kolekcję elementów bez zachowania kolejności i bez duplikatów.

Na początku stworzona jest klasa o nazwie `"LiczenieUnikatowychLiczb"`. W bloku kodu klasy stworzony jest obiekt klasy `Scanner`, który służy do wczytywania danych od użytkownika z klawiatury.

Następnie tworzony jest obiekt klasy `HashSet`, która służy do przechowywania unikalnych elementów. Wypisywany jest komunikat `"Podaj liczby całkowite (wprowadź 0, aby zakończyć):"`.

Wczytywana jest pierwsza liczba od użytkownika za pomocą metody `"nextInt"` klasy `Scanner` i przypisywana do zmiennej `"liczba"`.

Następnie stworzona jest pętla `"while"`, która będzie się wykonywać tak długo, aż użytkownik wprowadzi 0.

Wewnątrz pętli liczba podana przez użytkownika jest dodawana do obiektu `HashSet` za pomocą metody `"add"`. Po wprowadzeniu każdej liczby, program pyta użytkownika o kolejną, dopóki ta będzie różna od zera. Po zakończeniu pętli wypisywana jest ilość unikalnych wartości, która znajduje się w obiekcie `HashSet`, za pomocą metody `"size"`.

W metodzie `main` tworzony jest nowy obiekt klasy `LiczenieUnikatowychLiczb` i przypisany do zmiennej `liczba`.

Ćwiczenie 3

Napisz program zliczający liczbę dziur binarnych. Dziura binarna to sekwencja 0 osadzona pomiędzy 1 w strukturze binarnej, np.:

- 001001000000010101 liczba dziur: 4
- 010 liczba dziur: 0
- 0001001 liczba dziur: 1

Kod programu

```
import java.util.Scanner;
public class Main
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Podaj liczbę w systemie binarnym: ");
        String binarna = scanner.nextLine();
        int liczbaDziur = 0;
        boolean wewnatrzDziury = false;

        for (int i = 0; i < binarna.length(); i++)
        {
            char znak = binarna.charAt(i);

            if (znak == '1')
            {
                if (wewnatrzDziury) //jeśli wewnatrzDziury jest true to zwiększ
                    liczbaDziur++;
                wewnatrzDziury = true;
            }
            else if (znak == '0')
            {
                if (wewnatrzDziury)
                {
                    continue;
                }
            }
            else
            {
                System.out.println("Podana wartość nie jest liczbą binarną!");
                return;
            }
        }

        System.out.println("Liczba dziur binarnych: " + liczbaDziur);
    }
}
```

Wynik kodu programu

Przykład 1

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe"
```

```
Podaj liczbę w systemie binarnym: 010101
```

```
Liczba dziur binarnych: 2
```

Przykład 2

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe" "-ja\
```

```
Podaj liczbę w systemie binarnym: 000010101000000
```

```
Liczba dziur binarnych: 2
```

Opis działania programu i wykorzystanych technik programistycznych

Kod programu pobiera od użytkownika liczbę w systemie binarnym, a następnie liczy, ile dziur ma ta liczba w swoim zapisie binarnym. Na początku stworzony jest obiekt klasy Scanner, który pozwoli na pobieranie danych od użytkownika z konsoli. Potem użytkownik jest proszony o podanie liczby w systemie binarnym, a wynik jest zapisywany do zmiennej binarna.

Następnie w programie stworzone są zmienne liczbaDziur i wewnatrzDziury. Zmienna liczbaDziur będzie przechowywać ilość dziur, a zmienna wewnatrzDziury będzie służyć do określania, czy algorytm jest wewnątrz dziury.

Potem w pętli for każdy znaku w liczbie binarnej jest sprawdzany. Na początku pętli for zmienna znak przyjmuje wartość kolejnego znaku w liczbie binarnej. Najpierw sprawdzane jest, czy znak jest równy 1. Jeśli tak, to pętla sprawdza, czy jest wewnątrz dziury (wewnatrzDziury = true). Jeśli tak, to zwiększa się liczbaDziur o 1 oraz zmienna wewnatrzDziury jest ustawiana na true.

Potem sprawdzany jest znak, czy jest równy 0. Jeśli tak i zmienna wewnatrzDziury jest ustawiona na true, to algorytm przechodzi do kolejnego znaku. W ten sposób pomija wszystkie zera, które znajdują się wewnątrz dziur.

Potem sprawdzane jest, czy znak nie jest ani 0, ani 1. Jeśli tak, to pętla kończy działanie i wyświetla komunikat o błędzie. Na koniec wyświetlana jest liczba dziur binarnych w liczbie, którą podał użytkownik.

Ćwiczenie 4

Napisz program, który dla zadanych zbiorów o wartościach wprowadzonych przez użytkownika wykona następujące operacje:

- Suma zbiorów
- Różnica zbiorów A-B
- Część wspólna zbiorów
- Różnica symetryczna zbiorów

Kod programu

```
import java.util.*;

public class Main
{
    public static void main(String[] args)
    {
        // Wczytanie zbiorów od użytkownika
        Scanner scanner = new Scanner(System.in);

        System.out.print("Podaj elementy zbioru A oddzielone spacjami: ");
        String pobierzZbiorA = scanner.nextLine();
        Set<String> zbiorA = new
HashSet<>(Arrays.asList(pobierzZbiorA.split(" ")));

        System.out.print("Podaj elementy zbioru B oddzielone spacjami: ");
        String pobierzZbiorB = scanner.nextLine();
        Set<String> zbiorB = new
HashSet<>(Arrays.asList(pobierzZbiorB.split(" ")));

        // Wykonanie operacji na zbiorach
        // Suma
        Set<String> suma = new HashSet<>(zbiorA);
        suma.addAll(zbiorB);
        // Różnica
        Set<String> roznica = new HashSet<>(zbiorA);
        roznica.removeAll(zbiorB);
```

```

//Cześć wspólna zbiorów
Set<String> czesc_wspolna = new HashSet<>(zbiorA);
czesc_wspolna.retainAll(zbiorB);

//Różnica symetryczna zbiorów
Set<String> roznica_symetryczna = new HashSet<>(zbiorA);
roznica_symetryczna.addAll(zbiorB);

Set<String> tmp = new HashSet<>(zbiorA);
tmp.retainAll(zbiorB);
roznica_symetryczna.removeAll(tmp);

// Wyświetlenie wyniku
//Suma
System.out.println("Suma zbiorów: " + suma);
//Różnica
System.out.println("Różnica zbiorów: " + roznica);
//Cześć wspólna zbiorów
System.out.println("Cześć wspólna zbiorów: " + czesc_wspolna);
//Różnica symetryczna zbiorów
System.out.println("Różnica symetryczna zbiorów: " +
roznica_symetryczna);
}
}

```

Wynik kodu programu

Przykład 1

```

"C:\Program Files\Java\jdk-17.0.5\bin\java.exe" "-javaagent:C:\Program
Podaj elementy zbioru A oddzielone spacjami: 1 2 3 4 5 6 78
Podaj elementy zbioru B oddzielone spacjami: 9 8 7 4 44 10
Suma zbiorów: [44, 1, 78, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Różnica zbiorów: [1, 78, 2, 3, 5, 6]
Cześć wspólna zbiorów: [4]
Różnica symetryczna zbiorów: [4, 1, 78, 2, 3, 5, 6, 7, 8, 9, 10]

```

Przykład 2

```

"C:\Program Files\Java\jdk-17.0.5\bin\java.exe" "-javaagent
Podaj elementy zbioru A oddzielone spacjami: 9 8 7 6 5 4
Podaj elementy zbioru B oddzielone spacjami: 1 2 3 4 5 6
Suma zbiorów: [1, 2, 3, 4, 5, 6, 7, 8, 9]
Różnica zbiorów: [7, 8, 9]
Cześć wspólna zbiorów: [4, 5, 6]
Różnica symetryczna zbiorów: [1, 2, 3, 7, 8, 9]

```

Opis działania programu i wykorzystanych technik programistycznych

W początku programu importowane są wszystkie klasy z pakietu java.util. W funkcji main() program najpierw tworzy obiekt Scanner, który jest używany do pobrania elementów dwóch zbiorów od użytkownika. Następnie, dla każdego zbioru, program pobiera elementy od użytkownika jako ciąg znaków i dzieli je na pojedyncze elementy za pomocą metody split() klasy String, a następnie konwertuje je na zbiór za pomocą konstruktora HashSet<> (Arrays.asList(pobierzZbiorA.split(" "))). Następnie program wykonuje na zbiorach operacje zbiorowe: sumę, różnicę, część wspólną i różnicę symetryczną, za pomocą metod dostępnych w klasie Set (interfejs implementowany przez HashSet), a następnie wyświetla wyniki za pomocą System.out.println().

Sumę zbiorów stworzona jest przez nowy zbiór, do którego dodaje wszystkie elementy zbioru A, a następnie wszystkie elementy zbioru B, korzystając z metody `addAll()`.

Aby obliczyć różnicę zbiorów, program tworzy nowy zbiór, do którego dodaje wszystkie elementy zbioru A, a następnie usuwa z niego wszystkie elementy zbioru B, korzystając z metody `removeAll()`.

Aby obliczyć części wspólnej zbiorów, program tworzy nowy zbiór, do którego dodaje tylko te elementy, które występują w obu zbiorach, korzystając z metody `retainAll()`.

W przypadku różnicy symetrycznej zbiorów, program tworzy nowy zbiór, do którego dodaje wszystkie elementy obu zbiorów, a następnie usuwa z niego elementy wspólne dla obu zbiorów, korzystając z metod `addAll()` i `removeAll()`.

Ćwiczenie 5

Napisz program, który sprawdzi czy podana liczba jest liczbą pierwszą.

Kod programu

```
import java.util.Scanner;

public class Main
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Podaj liczbę: ");
        int liczba = scanner.nextInt();

        boolean czyPierwsza = true;

        if (liczba <= 1)
        {
            czyPierwsza = false;
        }

        for (int i = 2; i < liczba; i++)
        {
            if (liczba % i == 0)
            {
                czyPierwsza = false;
                break;
            }
        }

        if (czyPierwsza)
        {
            System.out.println(liczba + " jest liczbą pierwszą.");
        }
        else
        {
            System.out.println(liczba + " nie jest liczbą pierwszą.");
        }
    }
}
```

Wynik kodu programu

Przykład 1

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe"
```

```
Podaj liczbę: 12345
```

```
12345 nie jest liczbą pierwszą.
```


Przykład 2

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe"
```

```
Podaj liczbę: 5
```

```
5 jest liczbą pierwszą.
```

Przykład 3

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe" "-
```

```
Podaj liczbę: 8
```

```
8 nie jest liczbą pierwszą.
```

Przykład 4

```
"C:\Program Files\Java\jdk-17.0.5\bin\java.exe"
```

```
Podaj liczbę: 271
```

```
271 jest liczbą pierwszą.
```

Opis działania programu i wykorzystanych technik programistycznych

Na początku programu utworzono obiekt klasy Scanner, który pozwala na pobranie danych od użytkownika. Następnie program prosi użytkownika o podanie liczby.

Wartość liczby jest zapisywana do zmiennej "liczba". Zmienna "czyPierwsza" jest ustawiona na "true", co oznacza, że program zakłada, że podana liczba jest liczbą pierwszą.

Następnie sprawdzane są dwa warunki.

Pierwszy warunek sprawdza czy liczba jest mniejsza lub równa 1. Jeśli liczba jest mniejsza lub równa 1, to jest ona uznawana za nieprawidłową i zmienna "czyPierwsza" ustawiana jest na "false".

W przeciwnym przypadku program przeprowadza pętlę for, która zaczyna się od liczby 2 (ponieważ 1 i liczby ujemne nie są liczbami pierwszymi) i kończy się na liczbie przed podaną liczbą (ponieważ jeśli liczba jest podzielna przez jakąś liczbę większą niż 1 i mniejszą niż ta liczba, to nie jest ona liczbą pierwszą).

W pętli program sprawdza, czy podana liczba jest podzielna przez wartość "i". Jeśli jest, to zmienna "czyPierwsza" jest ustawiana na "false" i pętla jest przerywana. Jeśli liczba nie jest podzielna przez żadną liczbę z przedziału od 2 do liczby-1, to zmienna "czyPierwsza" pozostaje równa "true".

Na końcu program wyświetla odpowiedni komunikat w zależności od wartości zmiennej "czyPierwsza". Jeśli zmienna jest równa "true", to wyświetlany jest komunikat, że podana liczba jest liczbą pierwszą. W przeciwnym przypadku program wyświetla komunikat, że podana liczba nie jest liczbą pierwszą.

Program sprawdza, czy liczba jest pierwsza przez dzielenie jej przez kolejne liczby naturalne od 2 do liczby-1. Jeśli liczba dzieli się bez reszty przez którąś z tych liczb, to nie jest ona liczbą pierwszą.

Wnioski

W środowisku Windows programowanie w języku Java polega na pisaniu kodu źródłowego i kompilacji kodu przy użyciu kompilatora Javy. Następnie, skompilowany kod może być uruchomiony w interpreterze Javy. Programy te korzystają z różnych konstrukcji językowych, takich jak pętle, instrukcje warunkowe, tablice, listy, zbiory, itp.