# Java 2. praktiskais darbs

## Roberts Legzdiņš P1-2

Praktiskie uzdevumi Java programmēšanas valodā

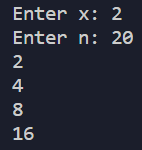
## 1. uzdevums

Ir uzdotie naturālie skaitļi n un x. Atrast un izvadīt uz ekrāna visas x skaitļa pakāpes, kuras nav lielākas kā n. Piemēram, ja n = 500, x = 5, tad atbilde būs 5, 25 un 125.

### Kods:

import java.util.Scanner;  
  
class Pirmais {  
  
 public static Scanner in = new Scanner(System.in);  
  
 public static int getNumber() {  
 int n = 0;  
  
 String nStr = in.next();  
  
 boolean valid = false;  
  
 while (!valid) {  
 try {  
 n = Integer.parseInt(nStr);  
  
 if (n > 0) {  
 valid = true;  
 } else {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 } catch (NumberFormatException e) {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 }  
  
 return n;  
 }  
  
 public static void main(String[] args) {  
 System.out.print("Enter x: ");  
 int x = getNumber();  
  
 System.out.print("Enter n: ");  
 int n = getNumber();  
  
 int xUp = 1;  
  
 while (xUp \* x < n) {  
 xUp = xUp \* x;  
  
 System.out.println(xUp);  
 }  
  
 in.close();  
 }  
}

### Rezultāts:



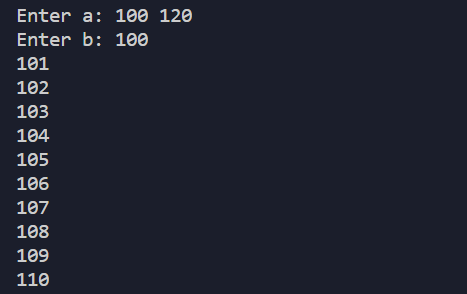
## 2. uzdevums

Ir dotie veselie skaitļi a un b. Izvadīt uz ekrāna visus skaitļus, kas atrodas diapazonā a…b, un kuru ieraksts satur vienu nulli. Piemēram, ja a = 108 un b = 130, tādi skaitļi būs 108, 109, 110, 120, 130.

### Kods:

import java.util.Scanner;  
  
public class otrais {  
  
 public static Scanner in = new Scanner(System.in);  
  
 public static int getNumber() {  
 int n = 0;  
  
 String nStr = in.next();  
  
 boolean valid = false;  
  
 while (!valid) {  
 try {  
 n = Integer.parseInt(nStr);  
  
 if (n >= 0) {  
 valid = true;  
 } else {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 } catch (NumberFormatException e) {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 }  
  
 return n;  
 }  
  
 public static void main(String[] args) {  
 System.out.print("Enter a: ");  
 int a = getNumber();  
  
 System.out.print("Enter b: ");  
 int b = getNumber();  
  
 while (a > b) {  
 System.out.println("a has to be < than b");  
 System.out.print("Enter b: ");  
 b = getNumber();  
 }  
  
 for (int counter = a; counter < b; counter++) {  
 if (String.valueOf(counter).contains("0")) {  
 System.out.println(counter);  
 }  
 }  
  
 in.close();  
 }  
}

### Rezultāts:



## 3. uzdevums

Sadalīt uzdoto naturāli skaitli n uz pirmreizinātājiem. Piemēram, skaitlim n=12 tāda sadalīšana būs 12 = 2·2·3

### Kods:

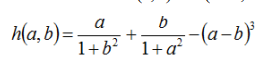
import java.util.Scanner;  
  
public class tresais {  
  
 public static Scanner in = new Scanner(System.in);  
  
 public static int getNumber() {  
 int n = 0;  
  
 String nStr = in.next();  
  
 boolean valid = false;  
  
 while (!valid) {  
 try {  
 n = Integer.parseInt(nStr);  
  
 if (n > 1) {  
 valid = true;  
 } else {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 } catch (NumberFormatException e) {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 }  
  
 return n;  
 }  
  
 public static int[] itt(int num) {  
 int[] val = new int[3];  
  
 for (int i = 2; i < num; i++) {  
 if (num % i == 0) {  
 val[0] = num / i;  
 val[1] = i;  
 val[2] = 1;  
 }  
 }  
  
 if(val[2] != 1){  
 val[0] = num;  
 val[1] = 1;  
 val[2] = -1;  
 }  
  
 System.out.print(val[0] + " ");  
  
 return val;  
 }  
  
 public static void main(String[] args) {  
 System.out.print("Enter n: ");  
 int n = getNumber();  
  
 int[] itt\_val = itt(n);  
  
 while (itt\_val[2] > 0) {  
 itt\_val = itt(itt\_val[1]);  
 }  
  
 in.close();  
 }  
}

### Rezultāts:

image

## 4. uzdevums

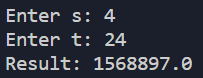
Aprēķināt izteiksmi: h(s, t) + h(s - t, s \* t) + h(1, 1);

 ir lietotāja definēta funkcija.

### Kods:

import java.util.Scanner;  
  
class ceturtais {  
 public static Scanner in = new Scanner(System.in);  
  
 public static int getNumber() {  
 int n = 0;  
  
 String nStr = in.next();  
  
 boolean valid = false;  
  
 while (!valid) {  
 try {  
 n = Integer.parseInt(nStr);  
  
 if (n > 1) {  
 valid = true;  
 } else {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 } catch (NumberFormatException e) {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 }  
  
 return n;  
 }  
  
 public static double h(int a, int b) {  
 double h = a / (1 + b \* b) + b / (1 + a \* a) - Math.pow((a - b), 3);  
  
 return h;  
 }  
  
 public static void main(String[] args) {  
 System.out.print("Enter s: ");  
 int s = getNumber();  
  
 System.out.print("Enter t: ");  
 int t = getNumber();  
  
 while (s == t) {  
 System.out.println("Numbers can't be equal");  
 System.out.print("Enter t: ");  
 t = getNumber();  
 }  
  
 double result = h(s, t) + h(s - t, s \* t) + h(1, 1);  
  
 System.out.print("Result: " + result);  
 }  
}

### Rezultāts:



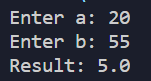
## 5. uzdevums

Uzrakstīt vesela tipa rekursīvu funkciju nod(a,b), kas meklē divu reālu skaitļu a un b lielāko kopīgo dalītāju, izmantojot Eiklīda algoritmu. Eiklīda algoritms ir efektīvs algoritms divu veselu skaitļu lielākā kopīgā dalītāja (LKD) atrašanai. Algoritms ir šāds: vispirms nepilni izdala lielāko skaitli ar mazāko un tad katrā nākamajā solī iepriekšējās darbības dalītāju dala ar iegūto atlikumu. Lai a un b veseli pozitivi skaitļi, kas nav vienādi nullei, un lai a>=b, tad: ja b = 0, tad LKD(a, 0) =a. Ja b nav vienāds 0, tad skaitļiem a, b un c, kur c ir atlikums no dalīšanas a uz b tiek izpildīts vienādība: LKD(a,b) = LKD(b,c).

### Kods:

import java.util.Scanner;  
  
public class piektais {  
 public static Scanner in = new Scanner(System.in);  
  
 public static int getNumber() {  
 int n = 0;  
  
 String nStr = in.next();  
  
 boolean valid = false;  
  
 while (!valid) {  
 try {  
 n = Integer.parseInt(nStr);  
  
 if (n > 1) {  
 valid = true;  
 } else {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 } catch (NumberFormatException e) {  
 System.out.print("Invalid integer, try again: ");  
 nStr = in.next();  
 }  
 }  
  
 return n;  
 }  
  
 public static void nod(double a, double b) {  
 double bint, sint, c;  
  
 bint = a;  
 sint = b;  
  
 if (b > a) {  
 bint = b;  
 sint = a;  
 }  
  
 c = bint % sint;  
  
 if(c == 0){  
 System.out.println("Result: " + sint);  
  
 return ;  
 }  
  
 nod(b, c);  
 }  
  
 public static void main(String[] args) {  
 System.out.print("Enter a: ");  
 int a = getNumber();  
  
 System.out.print("Enter b: ");  
 int b = getNumber();  
  
 nod(a, b);  
 }  
}

### Rezultāts:



## Secinājumi

Nebija lielas aizķeršanās. Radās nelielas neskaidrības, bet pēc kārtīgas iedziļināšanās, tās pazuda. Manuprāt uzdevumus izpildīju veiksmīgi.