LAPORAN PRAKTIKUM BASIS DATA LANJUT

SINTAKS PERULANGAN 1

Oleh:

MOCHAMMAD SYAIFUDDIN ZUHRI NIM. 1941720013



PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

29 OKTOBER 2019

# JOBSHEET 6

# SINTAKS PERULANGAN 1

## Kompetensi

Mahasiswa memahami serta mampu membuat program dalam bahasa Java menggunakan instruksi perulangan.

## Alat Dan Bahan:

1. PC/Laptop
2. JDK
3. Text editor (Sublime)

## Ulasan Teori:

*Loop* adalah suatu blok atau kelompok instruksi yang dilaksanakan secara berulang-ulang. Perulangan yang disebut juga repetition akan membuat efisiensi proses dibandingkan jika dioperasikan secara manual.

Perulangan yang dijelaskan pada *jobsheet* ini adalah :

* Perulangan dengan **for**
* Perulangan dengan **while**
* Perulangan dengan **do-while**

Penulisan sintaks ketiga jenis perulangan tersebut dibedakan sebagai berikut :

do {

statement;

…

} while(*kondisi*);

while(*kondisi*) {

statement;

…

}

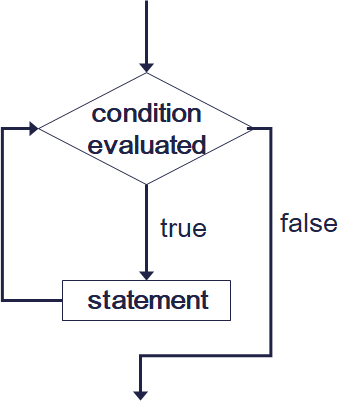
for(*inisialisasi; kondisi; update*) {

statement;

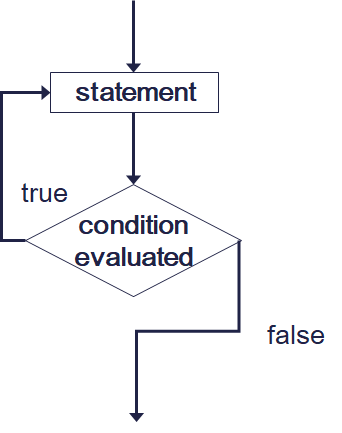
…

}

Ketiga jenis *loop* tersebut sama-sama memiliki kondisi yang merupakan batasan suatu perulangan dilakukan. Cara kerja *loop* menggunakan **for** dan **while** dijelaskan pada Gambar 8.1. Batasan yang menjadi kondisi suatu perulangan didefinisikan dulu di awal, kemudian dilanjutkan dengan *statement* yang harus di-*looping*. Berbeda dengan **for** dan **while**, Gambar 8.2 menjelaskan tentang alur perulangan menggunakan **do-while**. Setelah inisialisasi dilakukan, akan di proses dulu *statement* yang harus dijalankan, baru kemudian dilakukan pembatasan *looping* dalam penulisan kondisi.



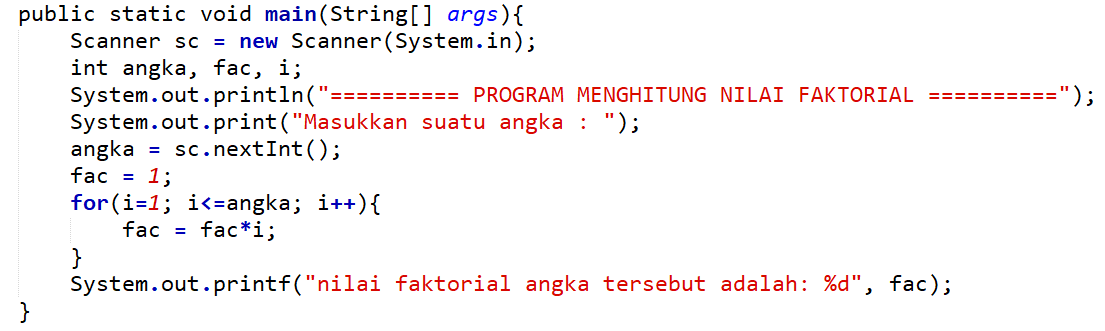
Gambar 6.1 *Flowchart* perulangan *for* dan *while*



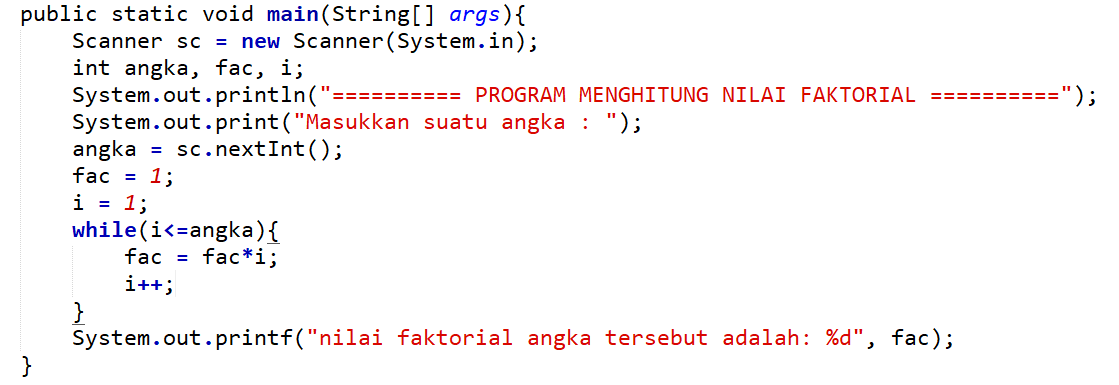
Gambar 6.2 *Flowchart* perulangan *do-while*

## Langkah Praktikum:

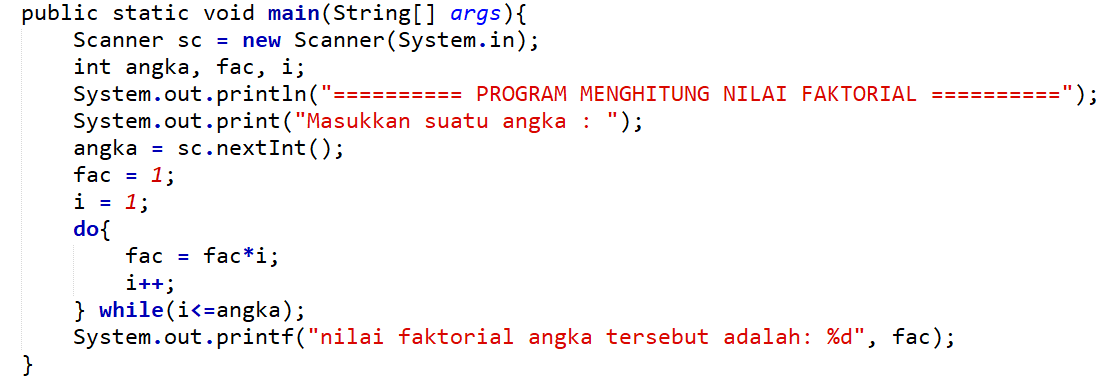
1. Tulis ulang program untuk melakukan perulangan sebagai berikut :
2. Perulangan dengan **for**



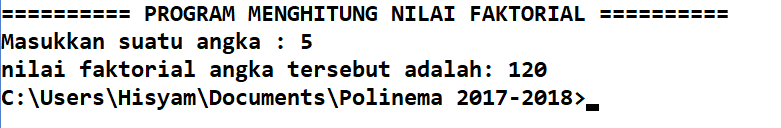
1. Perulangan dengan **while**



1. Perulangan dengan **do-while**

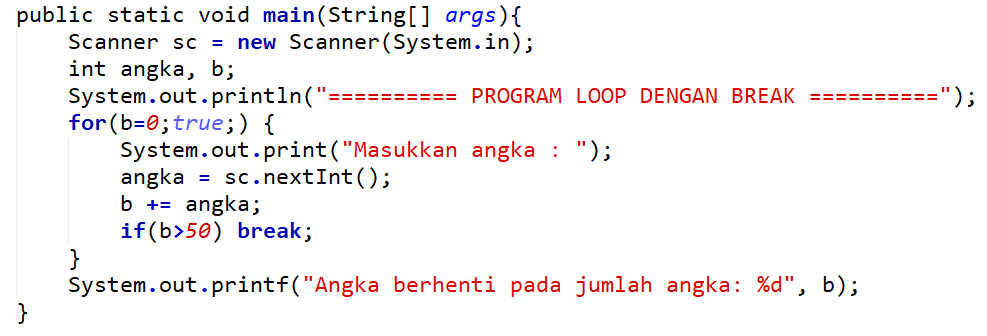


1. Cocokkan hasil *running* program yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut ini?

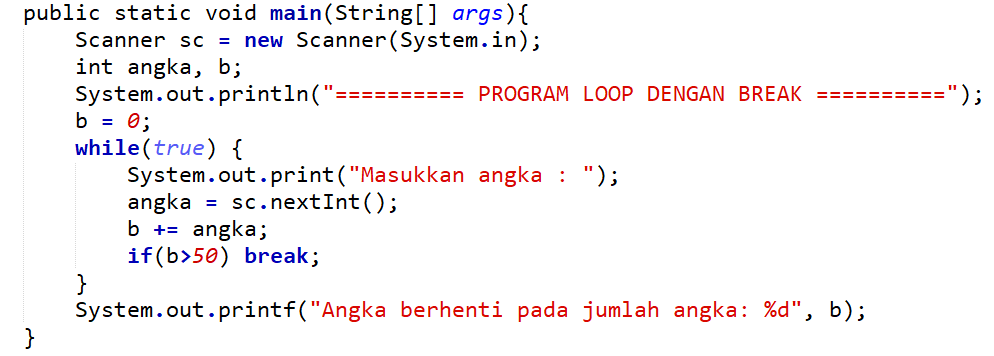


3. Salinlah program perulangan dengan menggunakan *break* berikut :

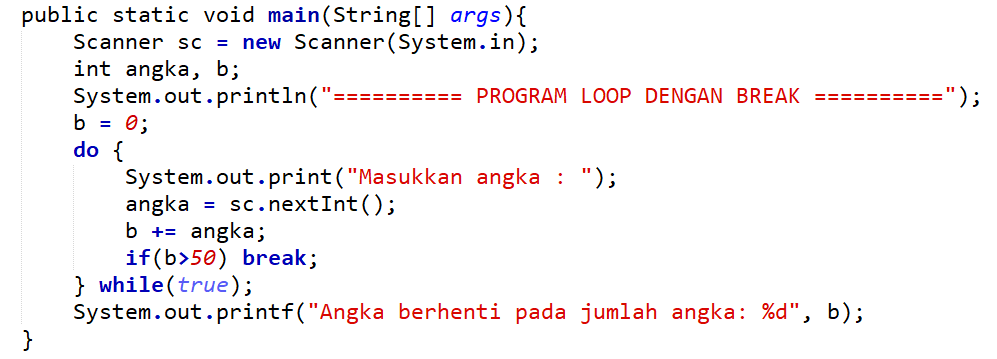
1. Perulangan dengan **for**



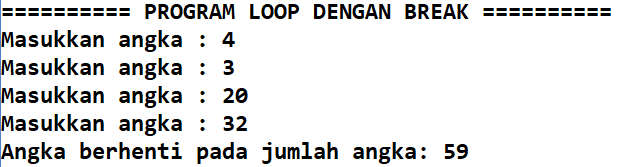
1. Perulangan dengan **while**



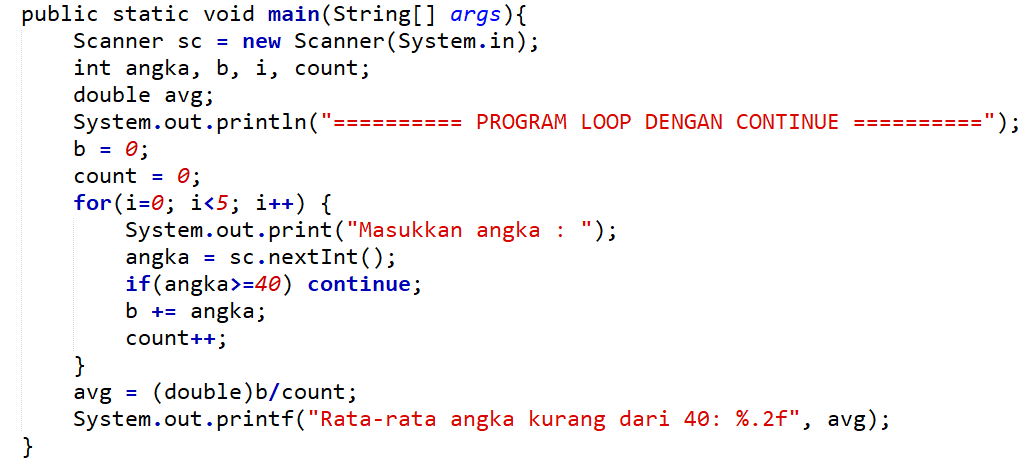
1. Perulangan dengan **do-while**



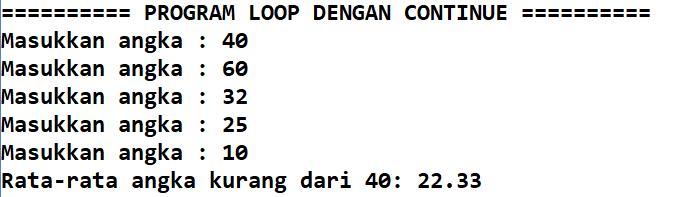
1. Cocokkan hasil *running* program *looping* menggunakan *break* yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut ini?



5. Salinlah program perulangan dengan menggunakan *continue* berikut :



1. Cocokkan hasil *running* program *looping* menggunakan *continue* yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut ini?



## Pertanyaan

1. Misalkan, Anda diminta membuat sebuah program Java yang meminta masukan sebuah bilangan bulat **n**. Kemudian, program menampilkan karakter **'\*'** di layar sebanyak **n kali**. Manakah di antara kedua potongan program di bawah ini yang lebih baik dan aman ? mengapa ?

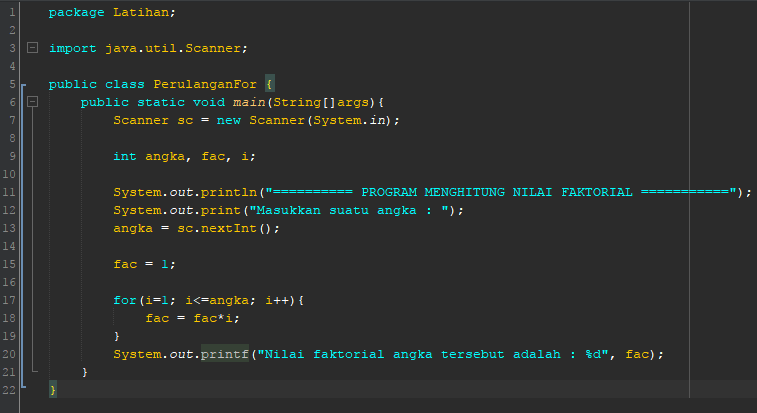
|  |  |
| --- | --- |
| /\* misal: masukan user n sudah ditampung di variabel integer n \*/  int i = 0;  while (i < n) {  System.out.print("\*");  i++;  } | /\* misal: masukan user n sudah ditampung di variabel integer n \*/  int i = 0;  while (i != n) {  System.out.print("\*");  i++;  } |

1. Apakah *output* dari ketiga potongan program di bawah ini:

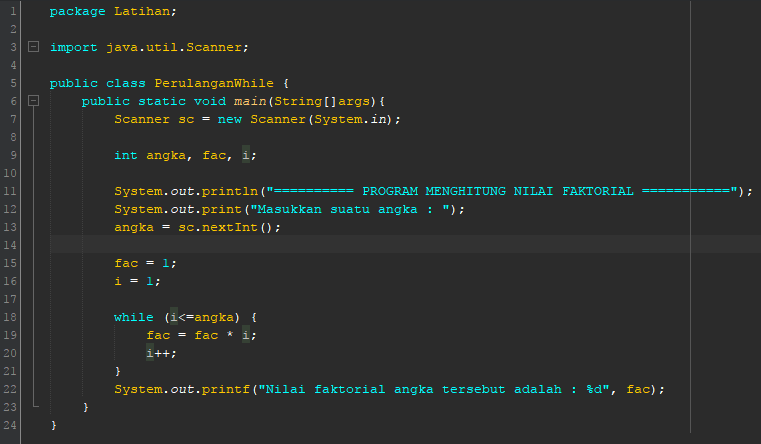
|  |  |  |
| --- | --- | --- |
| int r = 1;  int i = 1;  int a = 2;  int n = 4;  while (i <= n) {  r = r \* a;  i++;  }  System.out.print(r); | int n = 7;  boolean stop = false;  int i = 1;  while (!stop) {  if (i >= n) {  stop = true;  } else {  if (i % 2 == 0) {  System.out.print(“#”);  } else {  System.out.print(“\*”);  }  i++;  }  } | int n = 1892;  int sum = 0;  while (n > 0) {  int digit = n % 10;  sum = sum + digit;  n = n / 10;  }  System.out.println(sum); |

**HASIL LATIHAN PRAKTIKUM**

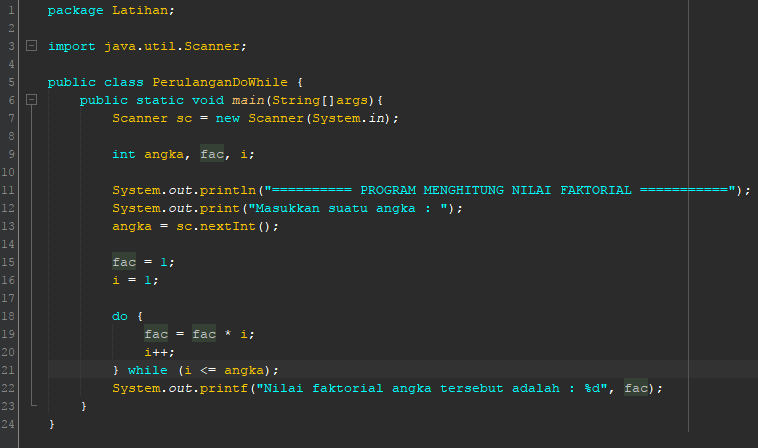
1. Perulangan biasa
   1. Perulangan **for**



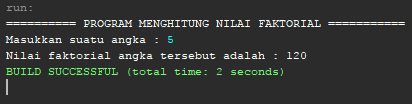
* 1. Perulangan **while**



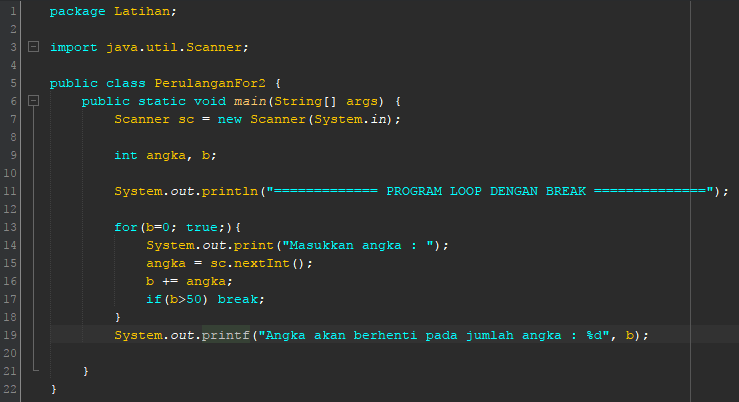
* 1. Perulangan **do-while**



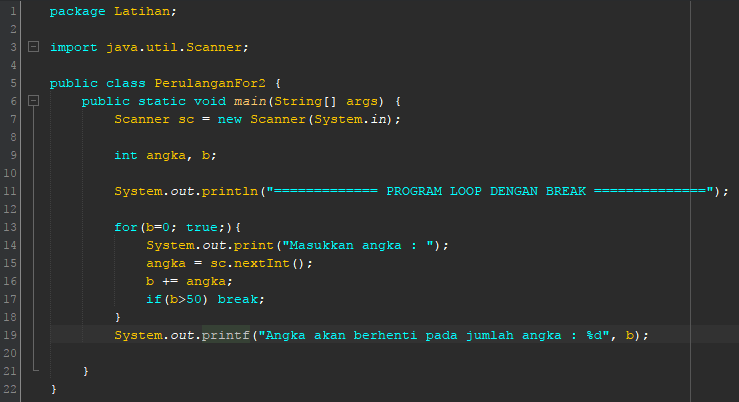
* 1. Hasil Program



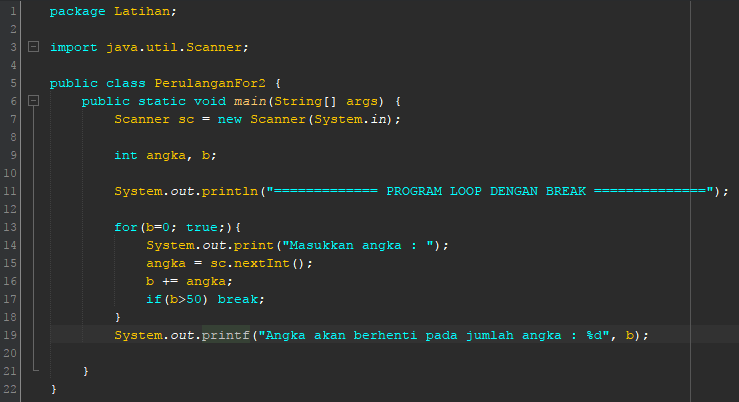
1. Perulangan dengan menggunakan break
   1. Perulangan **for**



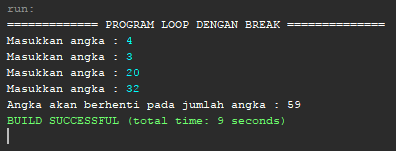
* 1. Perulangan **while**



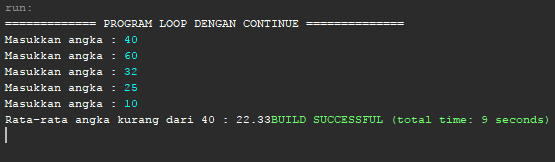
* 1. Perulangan **do-while**



* 1. Hasil Program



1. Perulangan dengan menggunakan continue



1. Pertanyaan
   1. Penggunaan kode program yang baik dan aman.

|  |  |
| --- | --- |
| /\* misal: masukan user n sudah ditampung di variabel integer n \*/  int i = 0;  while (i < n) {  System.out.print("\*");  i++;  } | /\* misal: masukan user n sudah ditampung di variabel integer n \*/  int i = 0;  while (i != n) {  System.out.print("\*");  i++;  } |
| 1 | 2 |

**Jawab :** Dari kedua potongan kode program diatas, penggunaan yang paling baik dan aman adalah kode program 1. Karena dalam program tersebut menggunakan operator <. Operator tersebut lebih aman karena akan menghentikan suatu perintah sesuai dengan kondisi yang dimasukkan.

* 1. Hasil program dibawah ini

|  |  |  |
| --- | --- | --- |
| int r = 1;  int i = 1;  int a = 2;  int n = 4;  while (i <= n) {  r = r \* a;  i++;  }  System.out.print(r); | int n = 7;  boolean stop = false;  int i = 1;  while (!stop) {  if (i >= n) {  stop = true;  } else {  if (i % 2 == 0) {  System.out.print(“#”);  } else {  System.out.print(“\*”);  }  i++;  }  } | int n = 1892;  int sum = 0;  while (n > 0) {  int digit = n % 10;  sum = sum + digit;  n = n / 10;  }  System.out.println(sum); |
| 1 | 2 | 3 |

**Jawab :**

|  |
| --- |
| **1** |
| D:\Kuliah\Mata Kuliah\Dasar Pemrograman\Semester 1\Tugas\JobSheet6\Capture\Contoh1.PNG |
|  |
| **2** |
| D:\Kuliah\Mata Kuliah\Dasar Pemrograman\Semester 1\Tugas\JobSheet6\Capture\Contoh 2.PNG |
|  |
| **3** |
| D:\Kuliah\Mata Kuliah\Dasar Pemrograman\Semester 1\Tugas\JobSheet6\Capture\Contoh3.PNG |

## Tugas

1. **(SumAvgGenap)** Buatlah program dengan menggunakan bahasa Java yang meminta masukan *user* sebuah bilangan bulat **N** (N > 0). Program kemudian menampilkan penjumlahan **N bilangan genap positif pertama (bilangan genap ≥ 0)**.

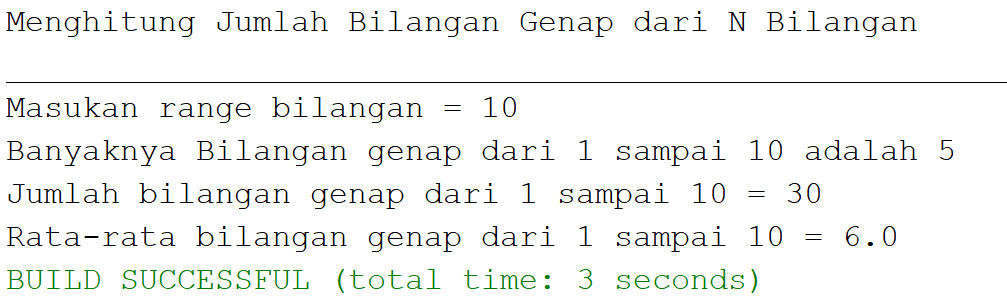
Contoh:

* Jika *user* memasukkan N = 10, program akan menghitung banyaknya jumlah bilangan positive di dalam range bilangan 1-10 kemudian menampilkan penjumlahan bilangan positive bilangan bilangan diantara 1-10 yaitu :

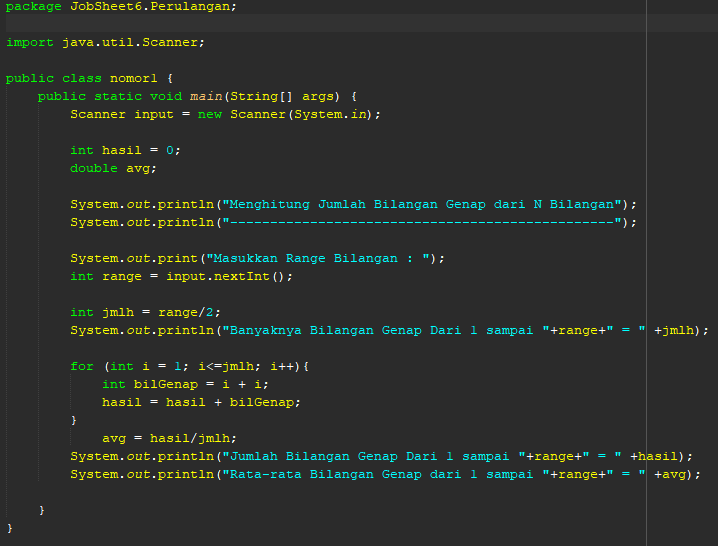
0 + 2 + 4 + 6 + 10 = 30.

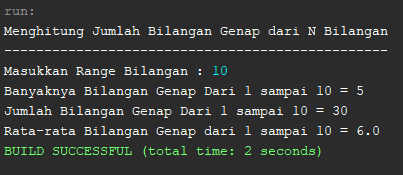
Setelah itu program akan menampilkan rata-rata dari bilangan positive yang telah dijumlahkan tadi.

* Contoh output program



Silakan Anda rancang sendiri untuk tampilan programnya

**Jawab :**



1. **(SumKGanjil)** Buatlah program dengan menggunakan bahasa Java yang meminta masukan *user* sebuah bilangan bulat **N** (N > 0). Program kemudian menampilkan penjumlahan **N bilangan ganjil positif pertama (bilangan genap ≥ 0)**.

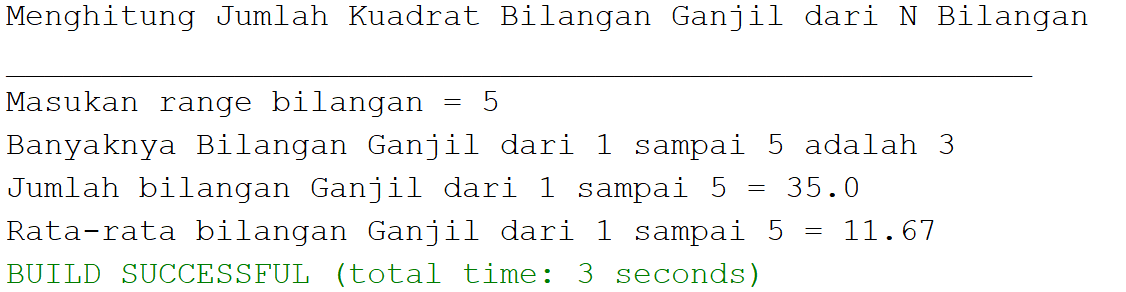
Contoh:

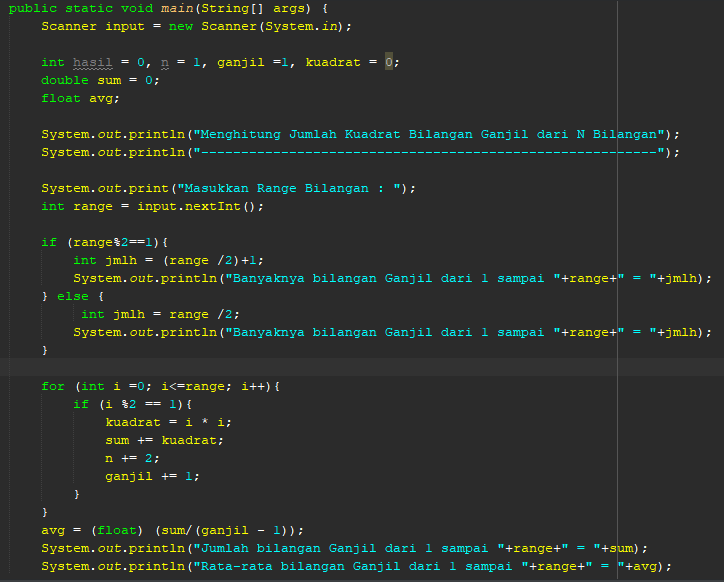
* Jika *user* memasukkan N = 5, program akan menghitung banyaknya jumlah bilangan positive di dalam range bilangan 1-5 kemudian menampilkan penjumlahan kuadrat bilangan ganjil diantara 1-5 yaitu :

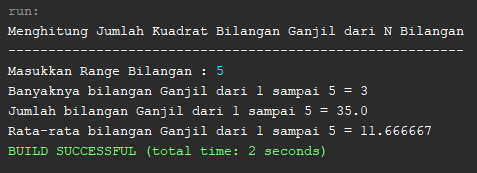
12 + 32 + 52 = 35.

Setelah itu program akan menampilkan rata-rata dari jumlah kuadrat bilangan negatif tersebut.

* Contoh output program



**Jawab :**

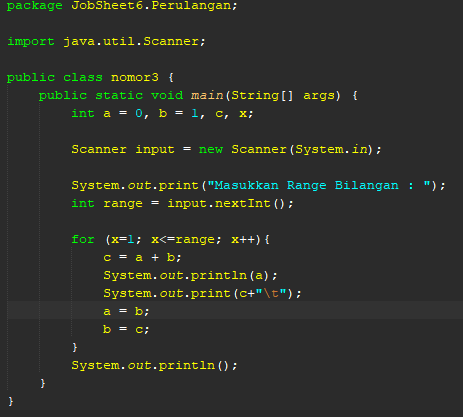


1. **(Fibonacci)** Buatlah sebuah program yang menampilkan deret bilangan sebagai berikut

0 1 1 2 3 5 8

Dimana rangkaian tersebut diperoleh dengan dimulai dari menjumlahkan angka 0+1=1, 1+1=2, 1+2=3 kemudian menjumlah 2+3=5, 5+8=13, dan seterusnya.



**Jawab :**

