**LAPORAN**

**PRAKTIKUM DASPRO PERTEMUAN 14**

****

**Nama :**

Ahmad Dzul Fadhli Hanna

**NIM :**

2341720106

**Kelas :**

TI-1H

**Absen :**

03

1. **TUJUAN**

* Mahasiswa memahami konsep fungsi rekursif
* Mahasiswa mampu mengimplementasikan fungsi rekursif dalam kode program

1. **PRAKTIKUM**
2. **Percobaan 1**

Codingan

/\*\*

\* Percobaan1

\*/

public class Percobaan1 {

static int faktorialRekursif(int n) {

if (n == 0) {

return (1);

} else {

return (n \* faktorialRekursif(n - 1));

}

}

static int faktorialIteratif(int n) {

int faktor = 1;

for (int i = n; i >= 1; i--) {

faktor = faktor \* i;

}

return faktor;

}

public static void main(String[] args) {

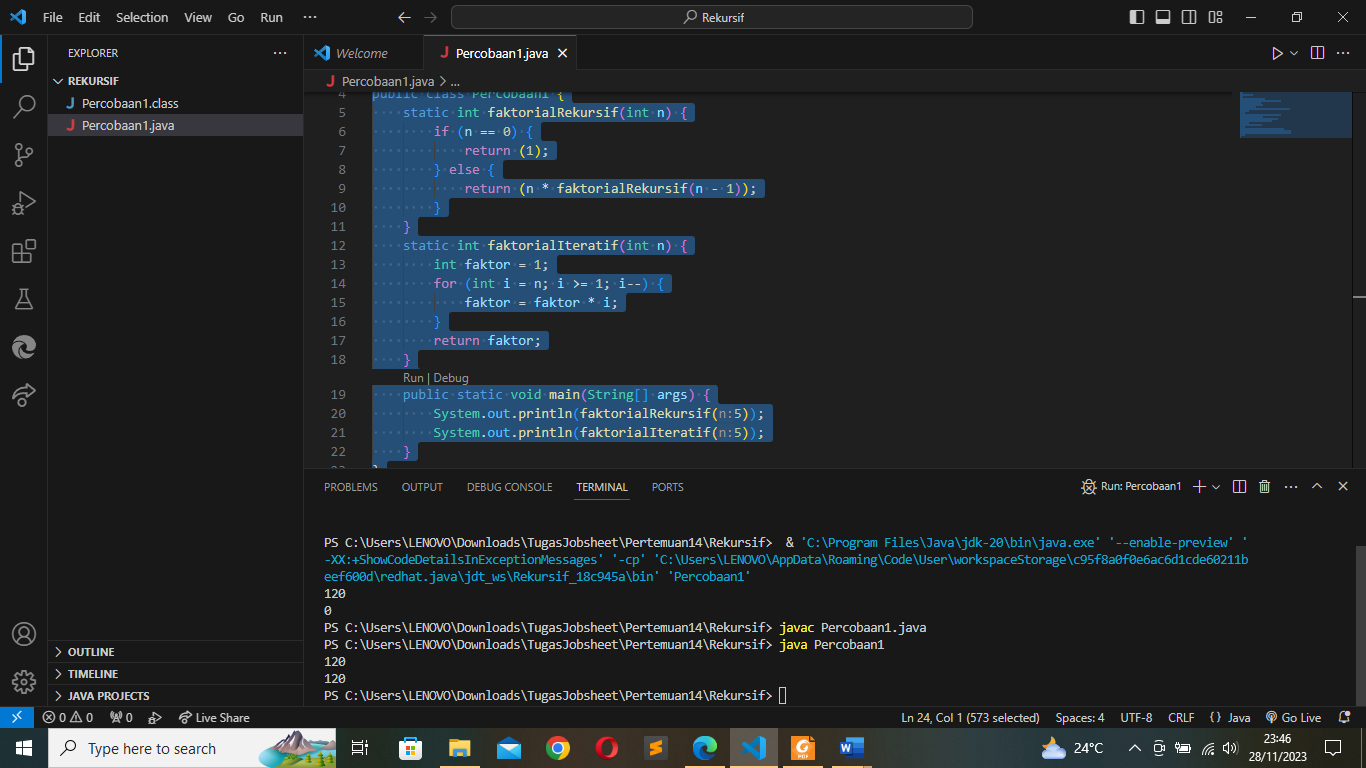
System.out.println(faktorialRekursif(5));

System.out.println(faktorialIteratif(5));

}

}

Hasil Running



**Pertanyaan**

1. Apa yang dimaksud dengan fungsi rekursif?

Fungsi yang memanggil dirinya sendiri.

1. Bagaimana contoh kasus penggunaan fungsi rekursif ?

- Menghitung nilai faktorial

- Menghhitung bilangan berpangkat

- Menghitung deret bilangan

- Melakukan pencarian

- Dll

1. Pada Percobaan1, apakah hasil yang diberikan fungsi faktorialRekursif() dan fungsi faktorialIteratif() sama? Jelaskan perbedaan alur jalannya program penggunaan fungsi rekursif dan fungsi iteratif!

faktorialRekursif()

* Memeriksa apakah n sama dengan 0
* Jika iya maka fungsi mengembalikan 1 sebagai basis rekursif
* Jika tidak, fungsi memanggil dirinya sendiri dengan parameter n-1 dan akan terus berulang hingga mencapai basis rekursif.
* Setiap kali fungsi memanggi dirinya sendiri, hasilnya akan dikalikan dengan nilai n-1 berulang-uang hingga hasinya mencapai basis rekursif.

faktorialIteratif()

* Inisialisasi variabel faktor dengan nilai 1 untuk menyimpan hasil perkalian faktorial
* Perulangan for untuk kondisi i sama dengan nilai n, akan berheti jika nilai i lebih besar atau sama dengan 1, dan i kurang satu untuk setiap perulangan yang dilakukan
* Variabe faktor dihitung dengan mengalikan faktor dengan niai i yang akan berulang sampai nilai i lebih besar atau sama dengan 1.
* Setelah perulangan selesai, nilai faktor yang sudah dihitung dikembalikan sebagai hasi faktorial.

1. **Percobaan 2**

Codingan

/\*\*

\* Percobaan2

\*/

import java.util.Scanner;

public class Percobaan2 {

static int hitungPangkat(int x, int y) {

if (y == 0) {

return(1);

}

else {

return(x \* hitungPangkat(x, y - 1));

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int bilangan, pangkat;

System.out.print("Bilangan yang ingin dihitung :");

bilangan = sc.nextInt();

System.out.print("Pangkat :");

pangkat = sc.nextInt();

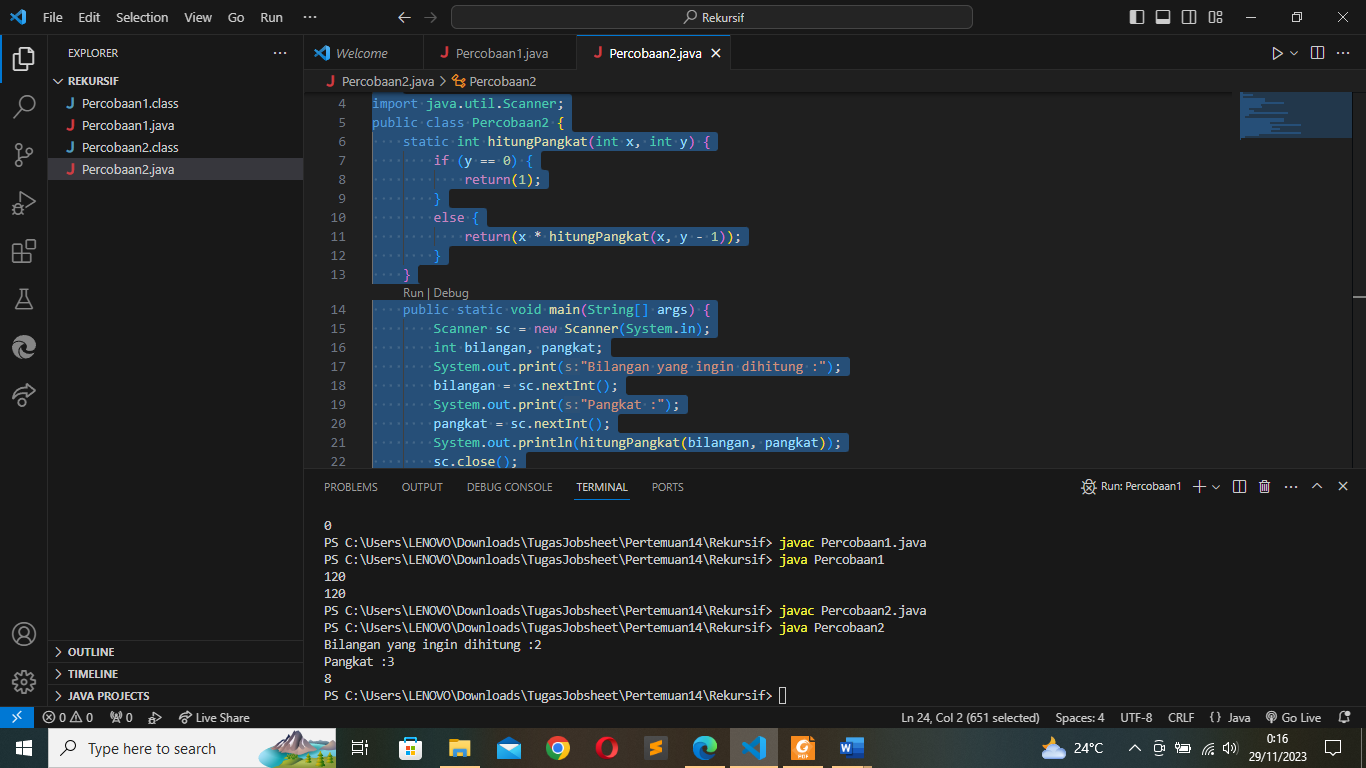
System.out.println(hitungPangkat(bilangan, pangkat));

sc.close();

}

}

Hasil Running



**Pertanyaan**

1. Pada Percobaan2, terdapat pemanggilan fungsi rekursif hitungPangkat(bilangan, pangkat) pada fungsi main, kemudian dilakukan pemanggilan hitungPangkat() secara berulangkali. Jelaskan sampai kapan proses pemanggilan fungsi akan dijalankan!

Proses pemanggilan fungsi akan terus berjalan hingga kondisi basis rekusrsif fungsi tercapai yaitu pada saat nilai y sama dengan 0 maka fungsi akan mengembalikan nilai 1 tanpa memanggil fungsi rekursif lagi.

1. Tambahkan kode program untuk mencetak deret perhitungan pangkatnya. Contoh : hitungPangkat(2,5) dicetak 2x2x2x2x2x1 = 32

Codingan

/\*\*

\* Percobaan2

\*/

import java.util.Scanner;

public class Percobaan2 {

static int hitungPangkat(int x, int y) {

if (y == 0) {

**System.out.print("1 ");**

return(1);

}

else {

**System.out.print(x + "\*");**

return(x \* hitungPangkat(x, y - 1));

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int bilangan, pangkat, **perhitungan**;

System.out.print("Bilangan yang ingin dihitung :");

bilangan = sc.nextInt();

System.out.print("Pangkat :");

pangkat = sc.nextInt();

**perhitungan = hitungPangkat(bilangan, pangkat);**

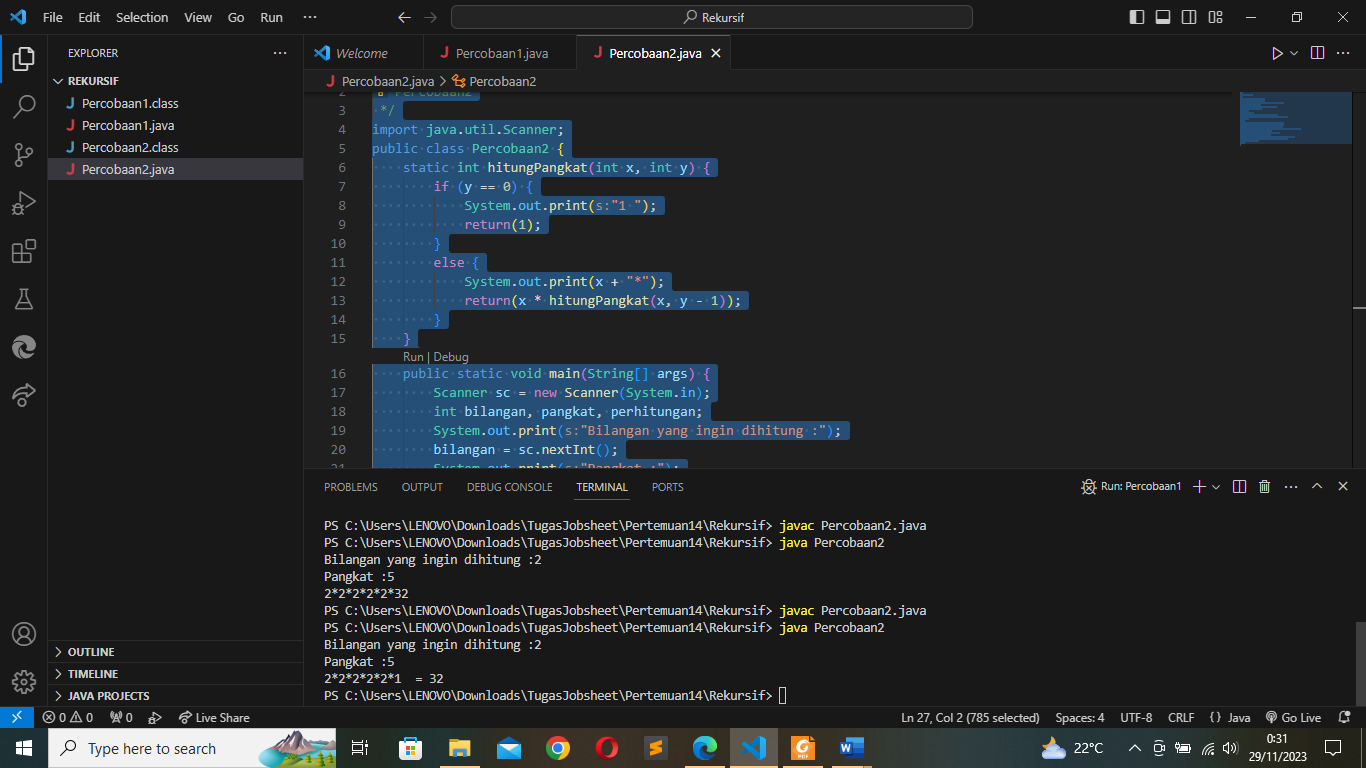
**System.out.println(" = " + perhitungan);**

sc.close();

}

}

Hasil Running



1. **Percobaan 3**

Codingan

/\*\*

\* Percobaan3

\*/

import java.util.Scanner;

public class Percobaan3 {

static double hitungLaba(double saldo, int tahun) {

if (tahun == 0) {

return (saldo);

}

else {

return (1.11 \* hitungLaba(saldo, tahun - 1));

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double saldoAwal;

int tahun;

System.out.print("Jumlah saldo awal : ");

saldoAwal = sc.nextDouble();

System.out.print("amanya investasi (tahun) : ");

tahun = sc.nextInt();

System.out.print("Jumlah saldo setelah " + tahun + "tahun : ");

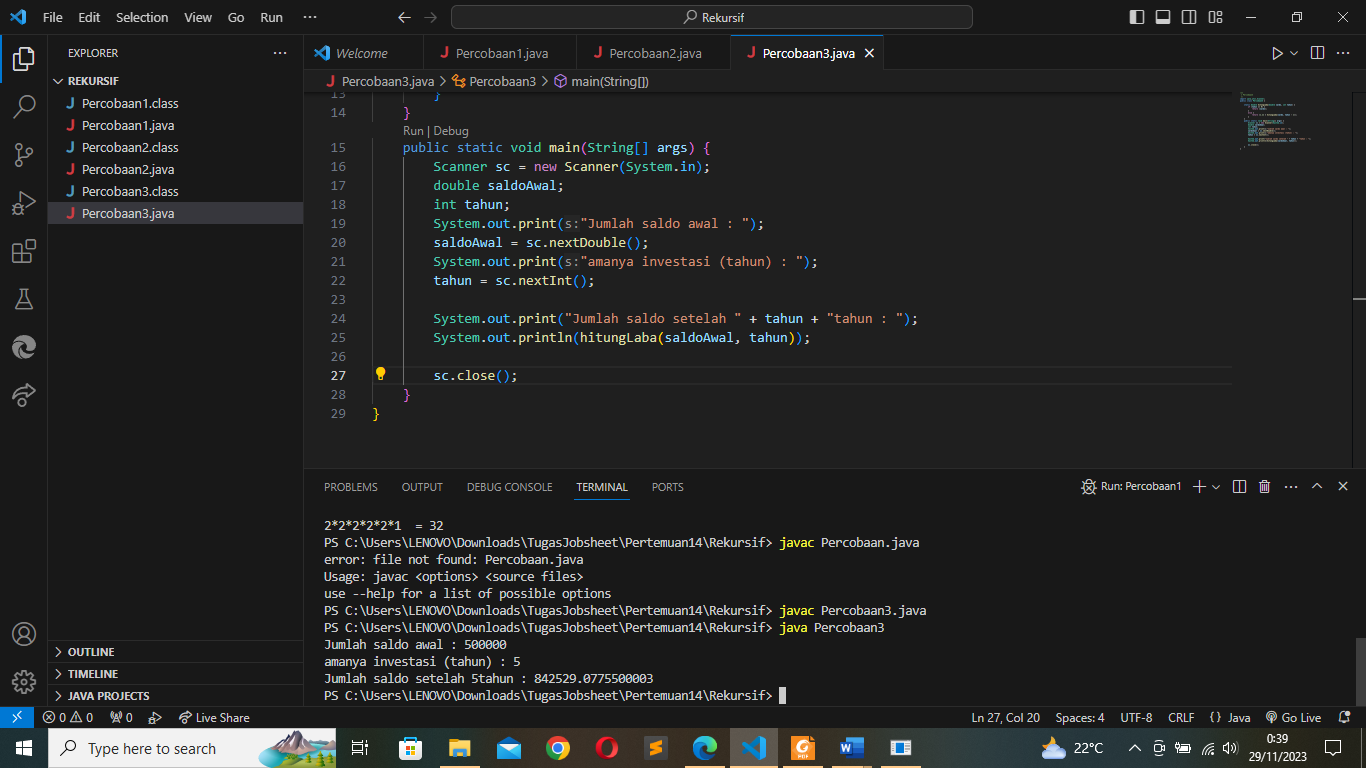
System.out.println(hitungLaba(saldoAwal, tahun));

sc.close();

}

}

Hasil Running



**Pertanyaan**

1. Pada Percobaan3, sebutkan blok kode program manakah yang merupakan “base case” dan “recursion call”!

static double hitungLaba(double saldo, int tahun) {

**//Base case**

if (tahun == 0) {

return (saldo);

}

**//Recursion call**

else {

return (1.11 \* hitungLaba(saldo, tahun - 1));

}

}

1. Jabarkan trace fase ekspansi dan fase subtitusi algoritma perhitungan laba di atas jika diberikan nilai hitungLaba(100000,3)

Trace Ekspansi:

hitungLaba (100000, 3)

Pemanggilan rekursif: 1.11 \* hitungLaba(100000, 2)

Pemanggilan rekursif: 1.11 \* hitungLaba(100000, 1)

Pemanggilan rekursif: 1.11 \* hitungLaba(100000, 0)

Base Case: Kondisi tahun == 0 tercapai, maka hasilnya adalah saldo (100000).

Trace Substitusi:

hitungLaba(100000, 3)

Pemanggilan rekursif: 1.11 \* hitungLaba(100000, 2)

Pemanggilan rekursif: 1.11 \* hitungLaba(100000, 1)

Pemanggilan rekursif: 1.11 \* hitungLaba(100000, 0)

Base Case: Menggantikan nilai hitungLaba(100000, 0) dengan saldo (100000).

Substitusi ke panggilan sebelumnya: 1.11 \* 100000

Substitusi ke panggilan sebelumnya: 1.11 \* (1.11 \* 100000)

Substitusi ke panggilan sebelumnya: 1.11 \* (1.11 \* (1.11 \* 100000))

Hasil Akhir: hitungLaba (100000, 3) sama dengan 1.11 \* (1.11 \* (1.11 \* 100000))

Hasil Akhir = 136763.10000000003

1. **TUGAS**
2. Buatlah program untuk menampilkan bilangan n sampai 0 dengan menggunakan fungsi rekursif dan fungsi iteratif. (DeretDescendingRekursif).

Codingan

import java.util.Scanner;

public class Tugas1 {

static int tampilkanRekursif(int n) {

if (n == 0) {

System.out.print(n + " ");

return(0);

}

else {

System.out.print(n + " ");

return(tampilkanRekursif(n-1));

}

}

static int tampilkanIteratif(int n) {

int mulai = n;

for (int i = mulai; i >= 0; i--) {

System.out.print(i + " ");

}

return mulai;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Masukkan nilai n :");

int n = sc.nextInt();

System.out.print("Rekursif :");

tampilkanRekursif(n);

System.out.print("Iteratif :");

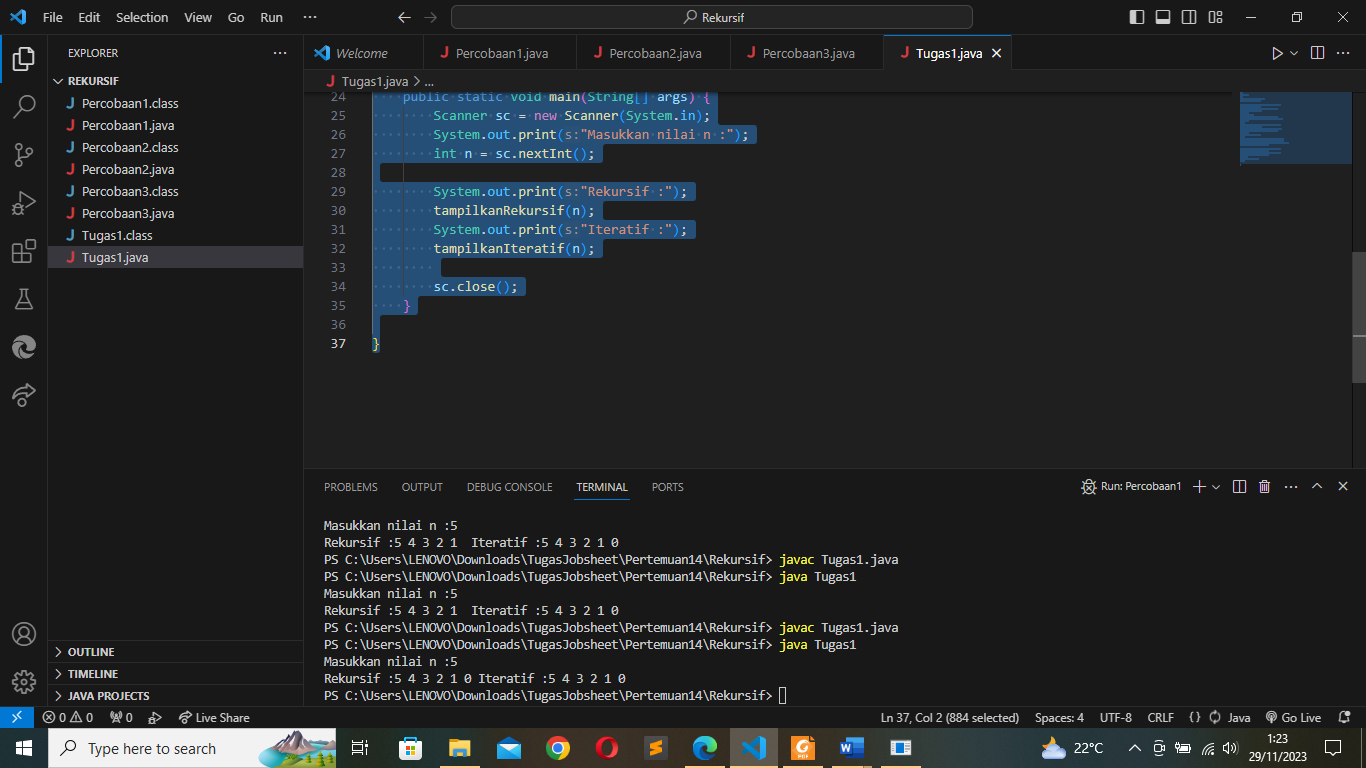
tampilkanIteratif(n);

sc.close();

}

}

Hasil Running



1. Buatlah program yang di dalamnya terdapat fungsi rekursif untuk menghitung penjumlahan bilangan. Misalnya f = 8, maka akan dihasilkan 1+2+3+4+5+6+7+8 = 36 (PenjumlahanRekursif)

Codingan

import java.util.Scanner;

public class Tugas2 {

static int penjumlahanRekursif(int f) {

if (f == 1) {

System.out.print("1 ");

return (1);

}

else {

int hasil = f + penjumlahanRekursif( f - 1);

System.out.print("+ " + f + " ");

return (hasil);

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Masukkan nilai f : ");

int f = sc.nextInt();

int hasil = penjumlahanRekursif(f);

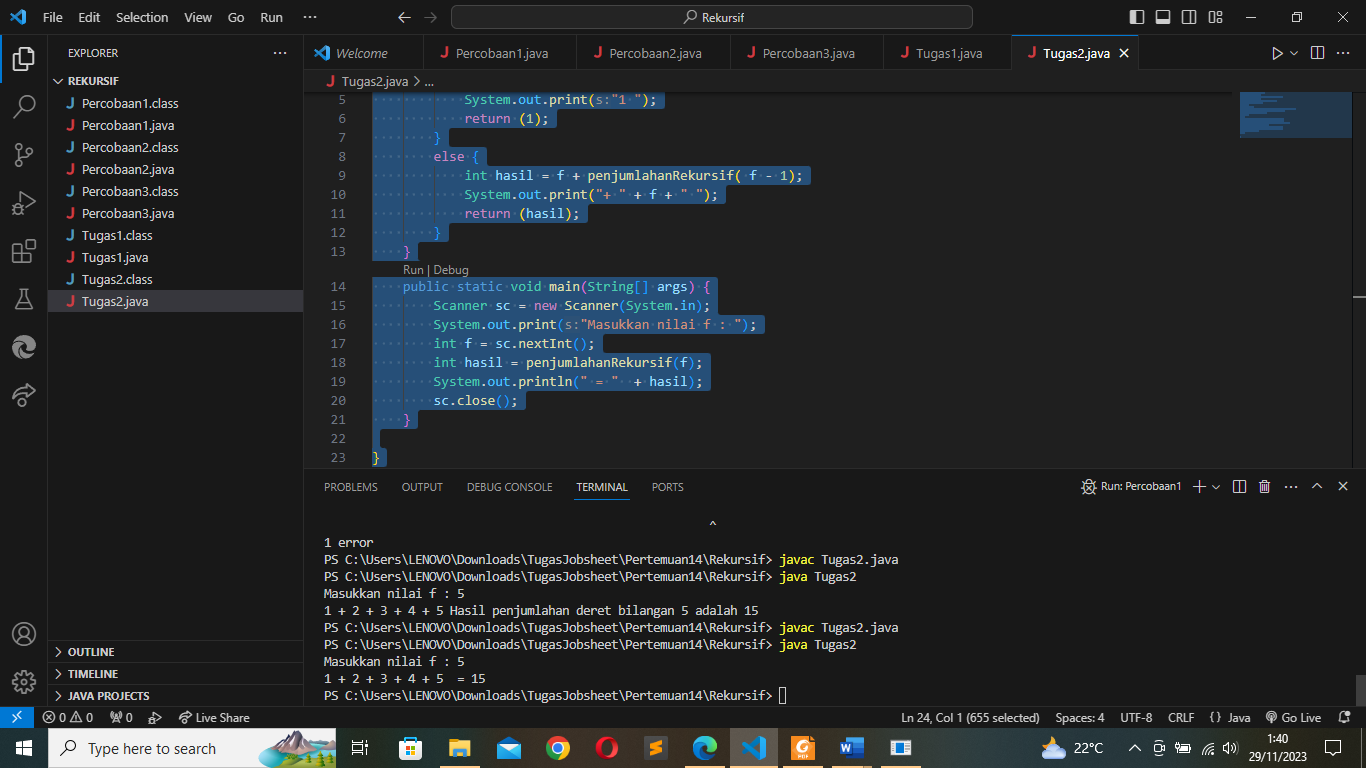
System.out.println(" = " + hasil);

sc.close();

}

}

Hasil Running



1. Buat program yang di dalamnya terdapat fungsi rekursif untuk mengecek apakah suatu bilangan n merupakan bilangan prima atau bukan. n dikatakan bukan bilangan prima jika ia habis dibagi dengan bilangan kurang dari n. (CekPrimaRekursif).

Codingan

import java.util.Scanner;

public class Tugas3 {

static int cekPrimaRekursif(int x, int y) {

if (x <= 1) {

return (0);

}

else if (y == 1) {

return (1);

}

else {

if (x % y == 0) {

return (0);

}

else {

return (cekPrimaRekursif(x, y - 1));

}

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Masukkan bilangan x : ");

int x = sc.nextInt();

int hasil = cekPrimaRekursif(x, x / 2);

if (hasil == 1) {

System.out.println(x + " merupakan bilangan prima");

}else{

System.out.println(x + " bukan bilangan prima");

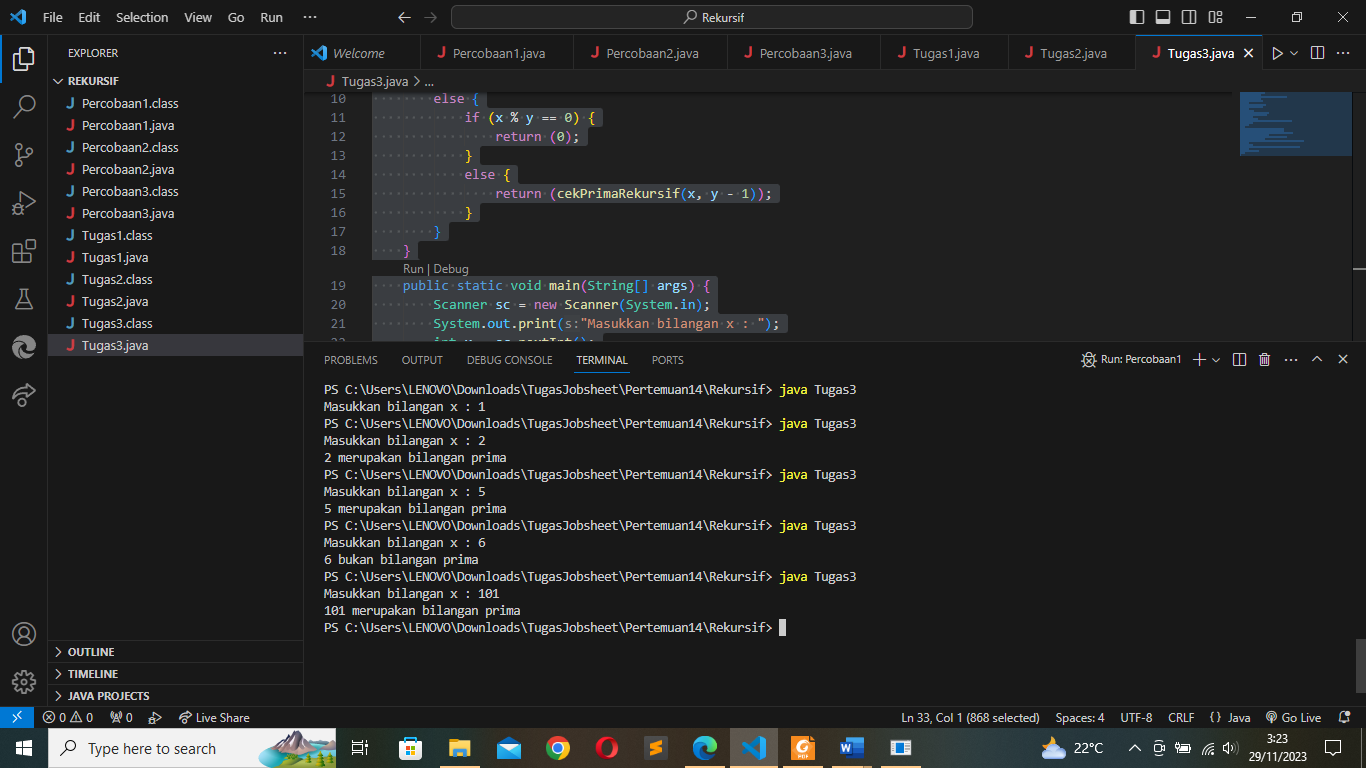
}

sc.close();

}

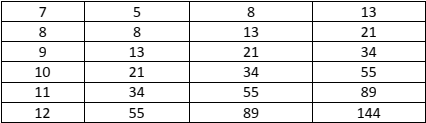
}

Hasil Running



1. Sepasang marmut yang baru lahir (jantan dan betina) ditempatkan pada suatu pembiakan. Setelah dua bulan pasangan marmut tersebut melahirkan sepasang marmut kembar (jantan dan betina). Setiap pasangan marmut yang lahir juga akan melahirkan sepasang marmut juga setiap 2 bulan. Berapa pasangan marmut yang ada pada akhir bulan ke-12? Buatlah programnya menggunakan fungsi rekursif! (Fibonacci). Berikut ini adalah ilustrasinya dalam bentuk tabel.





Codingan

public class Tugas4 {

static int fibonacciMarmut(int bulan) {

if (bulan <= 1) {

return bulan;

} else {

return fibonacciMarmut(bulan - 1) + fibonacciMarmut(bulan - 2);

}

}

public static void main(String[] args) {

int banyakMarmut = fibonacciMarmut(12);

System.out.println("Pada akhir bulan ke-12 banyak marmut adalah : " + banyakMarmut);

}

}

Hasil Running

