**TUGAS 5**

**MATA KULIAH ALGORITMA DAN DASAR PEMROGRAMAN SEMESTER GENAPTAHUN AKADEMIK 2023-2024**

****

**Oleh:**

**DEVA FAROIDL**

**4123004**

**PROGRAM STRATA-1 JURUSAN SISTEM INFORMASI**

**UNIVERSITAS PESANTREN TINGGI DARUL 'ULUM JOMBANG**

**2024**

Laporan tugas

Latihan 1:

Source code

public class Kalender {

private int tanggal;

private int bulan;

private int tahun;

// Konstruktor

public Kalender(int tanggal) {

this(tanggal, LocalDate.now().getMonthValue(), LocalDate.now().getYear());

}

public Kalender(int bulan, int tahun) {

this(1, bulan, tahun);

}

public Kalender(int tanggal, int bulan, int tahun) {

if (tanggal <= 0 || tanggal > getDaysFromMonth(bulan, tahun)) {

throw new IllegalArgumentException("Tanggal tidak valid.");

}

this.tanggal = tanggal;

this.bulan = bulan;

this.tahun = tahun;

}

// Getter

public int getTanggal() {

return tanggal;

}

public int getBulan() {

return bulan;

}

public int getTahun() {

return tahun;

}

// Setter

public void setTanggal(int tanggal) {

if (tanggal <= 0 || tanggal > getDaysFromMonth(bulan, tahun)) {

throw new IllegalArgumentException("Tanggal tidak valid.");

}

this.tanggal = tanggal;

}

public void setBulan(int bulan) {

if (bulan < 1 || bulan > 12) {

throw new IllegalArgumentException("Bulan tidak valid.");

}

this.bulan = bulan;

}

public void setTahun(int tahun) {

if (tahun < 0) {

throw new IllegalArgumentException("Tahun tidak valid.");

}

this.tahun = tahun;

}

// Method

public int getDaysFromMonth(int month, int year) {

return LocalDate.of(year, month, 1).lengthOfMonth();

}

public static String getTime(Kalender kal) {

String tmp;

tmp = kal.getTanggal() + "-" +

kal.getBulan() + "-" +

kal.getTahun();

return tmp;

}

@Override

public String toString() {

return getTime(this);

}

}

class TestKalender {

public static void main(String args[]) {

Kalender cal = new Kalender(8);

System.out.println("Waktu awal : " + getTime(cal) );

cal.setTanggal(9);

System.out.println("1 hari setelah waktu awal : " + getTime(cal));

cal = new Kalender(6, 2003);

System.out.println("Waktu berubah : "+ getTime(cal));

cal.setBulan(7);

System.out.println("1 bulan setelah itu : "+ getTime(cal));

cal = new Kalender(20, 10, 2004);

System.out.println("Waktu berubah : "+ getTime(cal) );

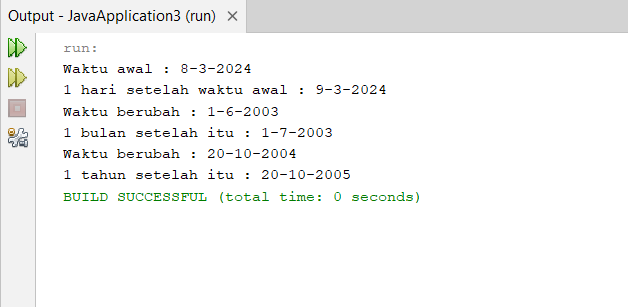
cal.setTahun(2005);

System.out.println("1 tahun setelah itu : "+ getTime(cal) );

}

}

Output:



Kelas Kalender memiliki method getTime yang didefinisikan untuk mengembalikan tanggal, bulan, dan tahun dalam bentuk string. Dalam kelas TestKalender, objek Kalender dibuat dengan tanggal, bulan, dan tahun yang berbeda-beda, dan setiap perubahan yang dilakukan pada objek Kalender diikuti oleh output dari method getTime.

Latihan 2

Source code:

public class Truk {

private double muatan;

private final double muatanMaks;

public Truk(double beratMaks) {

this.muatan = 0.0;

this.muatanMaks = beratMaks;

}

public double getMuatan() {

return muatan;

}

public double getMuatanMaks() {

return muatanMaks;

}

public boolean tambahMuatan(double berat) {

if (muatan + berat > muatanMaks) {

return false;

}

muatan += berat;

return true;

}

public double newtsToKilo(double berat) {

return berat / 9.8;

}

public double kiloToNewts(double berat) {

return berat \* 9.8;

}

}

class TesTugas2 {

public static void main(String args[]) {

boolean status;

Truk truk = new Truk(900);

System.out.println("Muatan maksimal = " + truk.getMuatanMaks());

status = truk.tambahMuatan(500.0);

System.out.println("Tambah muatan : 500");

if (status)

System.out.println("Ok");

else

System.out.println("Gagal");

status = truk.tambahMuatan(300.0);

System.out.println("Tambah muatan : 300");

if (status)

System.out.println("Ok");

else

System.out.println("Gagal");

status = truk.tambahMuatan(150.0);

System.out.println("Tambah muatan : 150");

if (status)

System.out.println("Ok");

else

System.out.println("Gagal");

status = truk.tambahMuatan(50.0);

System.out.println("Tambah muatan : 50");

if (status)

System.out.println("Ok");

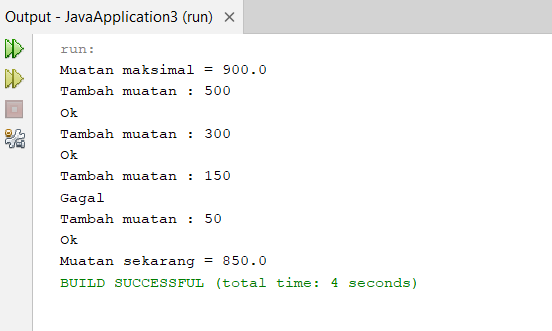
else

System.out.println("Gagal");

System.out.println("Muatan sekarang = " + truk.getMuatan());

}

}



Dalam kelas Truk, konstruktor yang didefinisikan membuat objek baru dengan muatan maksimal yang diberikan. Method getMuatan, getMuatanMaks, tambahMuatan, newtsToKilo, dan kiloToNewts didefinisikan sesuai dengan class diagram.

Dalam kelas TesTugas2, objek Truk dibuat dengan muatan maksimal 900. Muatan ditambah dengan berturut-turut 500, 300, 150, dan 50. Setiap tambah muatan diikuti oleh pesan "Ok" jika berhasil atau "Gagal" jika gagal.

Tugas 1:

Source code:

class MultiTabungan {

// saldo disimpan dalam IDR

private int saldo;

// Konstruktor

MultiTabungan(int initsaldo) {

this.saldo = initsaldo;

}

// Mendapatkan saldo dalam IDR

int getSaldo() {

return saldo;

}

// Menyimpan jumlah dalam mata uang yang dipilih

void simpanUang(String mataUang, int jumlah) {

if (mataUang.equalsIgnoreCase("USD")) {

saldo += 9000 \* jumlah;

} else if (mataUang.equalsIgnoreCase("AUD")) {

saldo += 10000 \* jumlah;

} else if (mataUang.equalsIgnoreCase("IDR")) {

saldo += jumlah;

} else {

System.out.println("Mata uang tidak valid.");

}

}

// Menarik jumlah dalam mata uang yang dipilih

boolean ambilUang(String mataUang, int jumlah) {

int jumlahIDR = 0;

if (mataUang.equalsIgnoreCase("USD")) {

jumlahIDR = 9000 \* jumlah;

} else if (mataUang.equalsIgnoreCase("AUD")) {

jumlahIDR = 10000 \* jumlah;

} else if (mataUang.equalsIgnoreCase("IDR")) {

jumlahIDR = jumlah;

} else {

System.out.println("Mata uang tidak valid.");

return false;

}

if (saldo >= jumlahIDR) {

saldo -= jumlahIDR;

return true;

} else {

System.out.println("Saldo tidak cukup.");

return false;

}

}

}

class TestMultiTabungan {

public static void main(String[] args) {

MultiTabungan rekening = new MultiTabungan(1000000);

rekening.simpanUang("AUD", 10);

System.out.println("Saldo: " + rekening.getSaldo() + " IDR");

if (rekening.ambilUang("IDR", 1000000)) {

System.out.println("Penarikan berhasil.");

}

rekening.simpanUang("USD", 10);

System.out.println("Saldo: " + rekening.getSaldo() + " IDR");

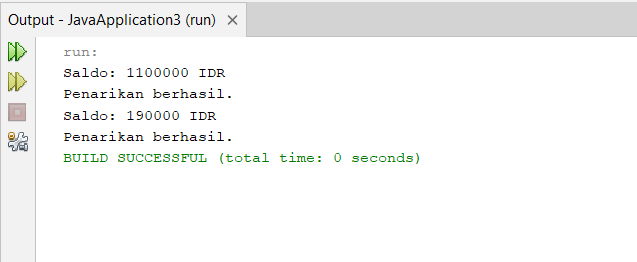
if (rekening.ambilUang("USD", 10)) {

System.out.println("Penarikan berhasil.");

}

}

}



Kode di atas mendefinisikan class MultiTabungan yang mewakili tabungan dengan saldo yang disimpan dalam mata uang IDR (Rupiah Indonesia). Class ini memiliki beberapa method, yaitu:

MultiTabungan(int initsaldo): Konstruktor untuk membuat objek baru dari class MultiTabungan dengan saldo awal yang diberikan dalam mata uang IDR.

int getSaldo(): Method untuk mendapatkan saldo tabungan saat ini dalam mata uang IDR.

void simpanUang(String mataUang, int jumlah): Method untuk menyimpan jumlah uang dalam mata uang yang dipilih (IDR, USD, atau AUD) ke dalam tabungan.

boolean ambilUang(String mataUang, int jumlah): Method untuk menarik jumlah uang dalam mata uang yang dipilih (IDR, USD, atau AUD) dari tabungan.

Kelas TestMultiTabungan menghasilkan output dengan melakukan beberapa transaksi pada objek MultiTabungan yang telah dibuat.

1. Membuat objek MultiTabungan dengan saldo awal 1.000.000 IDR.
2. Menyimpan uang sebesar 10 AUD ke dalam tabungan.
3. Menampilkan saldo tabungan saat ini dalam mata uang IDR.
4. Menarik uang sebesar 1.000.000 IDR dari tabungan.
5. Menampilkan saldo tabungan saat ini dalam mata uang IDR.
6. Menyimpan uang sebesar 10 USD ke dalam tabungan.
7. Menarik uang sebesar 10 USD dari tabungan.