**NAMA : WILLY ACHMAD N**

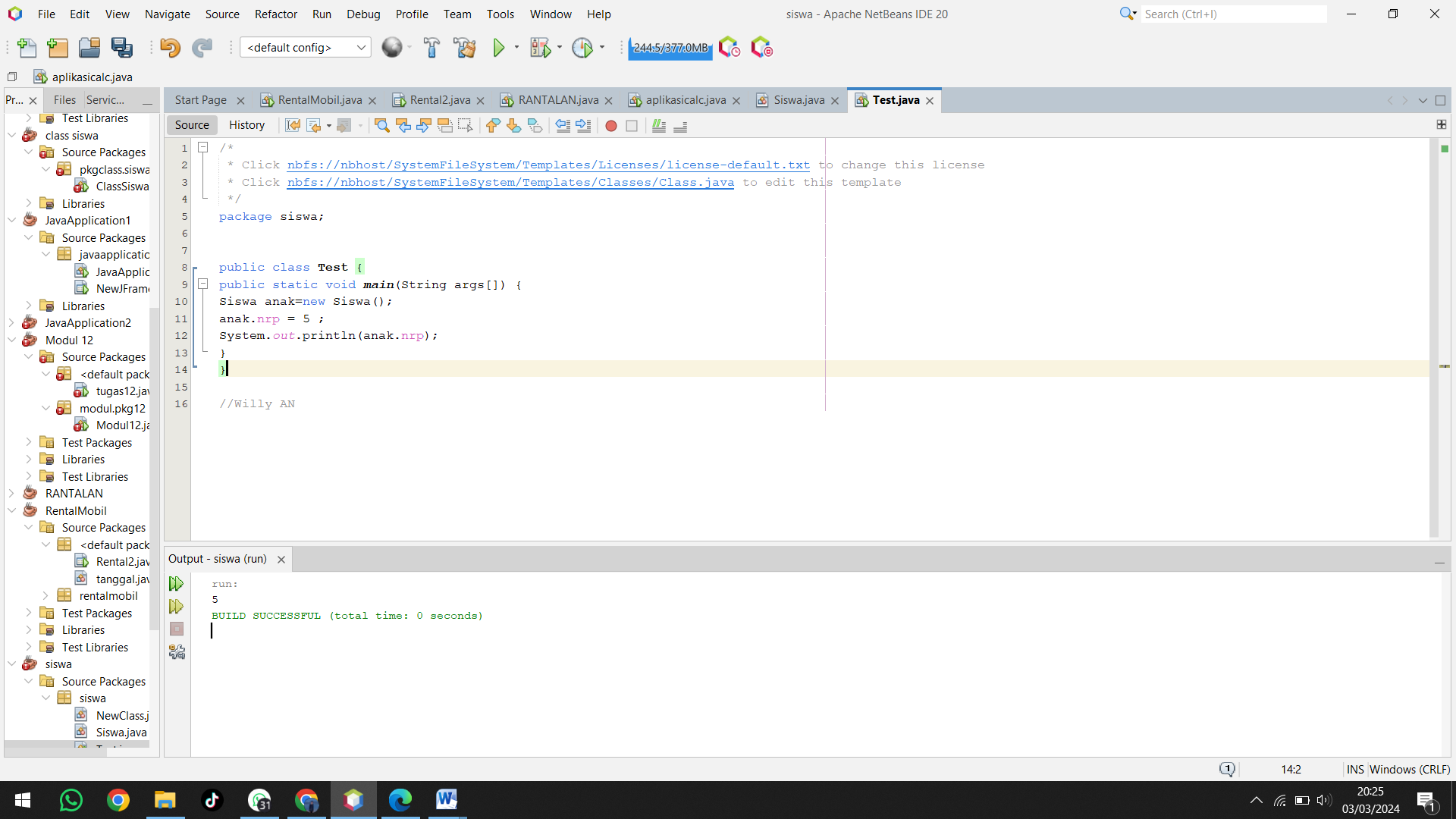
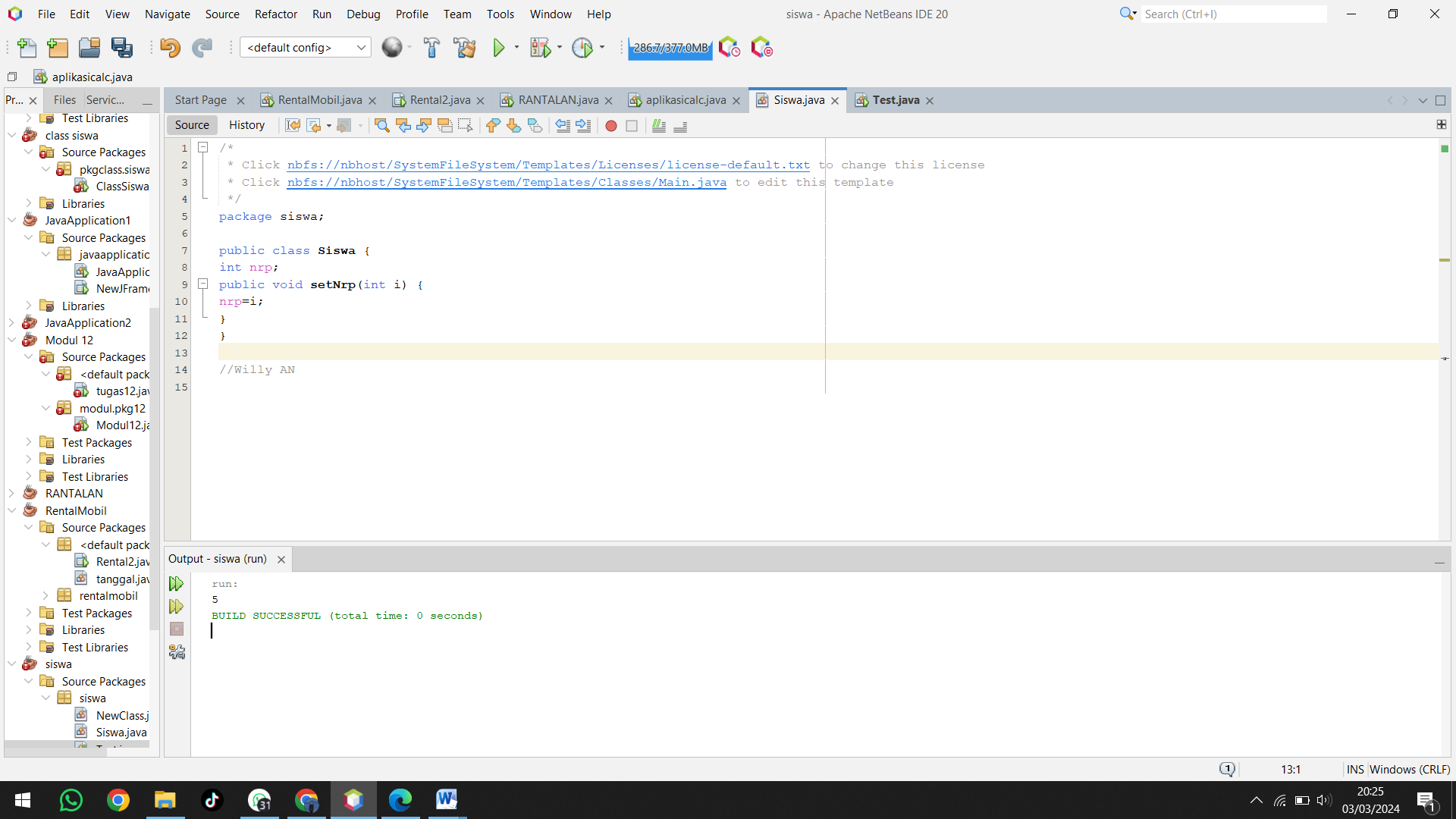
**NIM : 4123024**

**KELAS : B**

**SEMESTER : 2**

**D. Percobaan**

**Nomor 1**



**package siswa;**

**public class Siswa {**

**int nrp;**

**public void setNrp(int i) {**

**nrp=i;**

**}**

**}**

**public class Test {**

**public static void main(String args[]) {**

**Siswa anak=new Siswa();**

**anak.nrp = 5 ;**

**System.out.println(anak.nrp);**

**}**

**}**

**Penjelasan: Program di atas adalah implementasi sederhana dari dua kelas Java, yaitu namasiswa dan siswa. Berikut adalah penjelasan fungsi programnya:**

**1. \*Deklarasi Kelas namasiswa:\***

**- Kelas ini memiliki satu atribut (int nrp), yang dapat menyimpan nomor registrasi siswa.**

**2. \*Kelas siswa:\***

**- Kelas utama yang berisi metode main.**

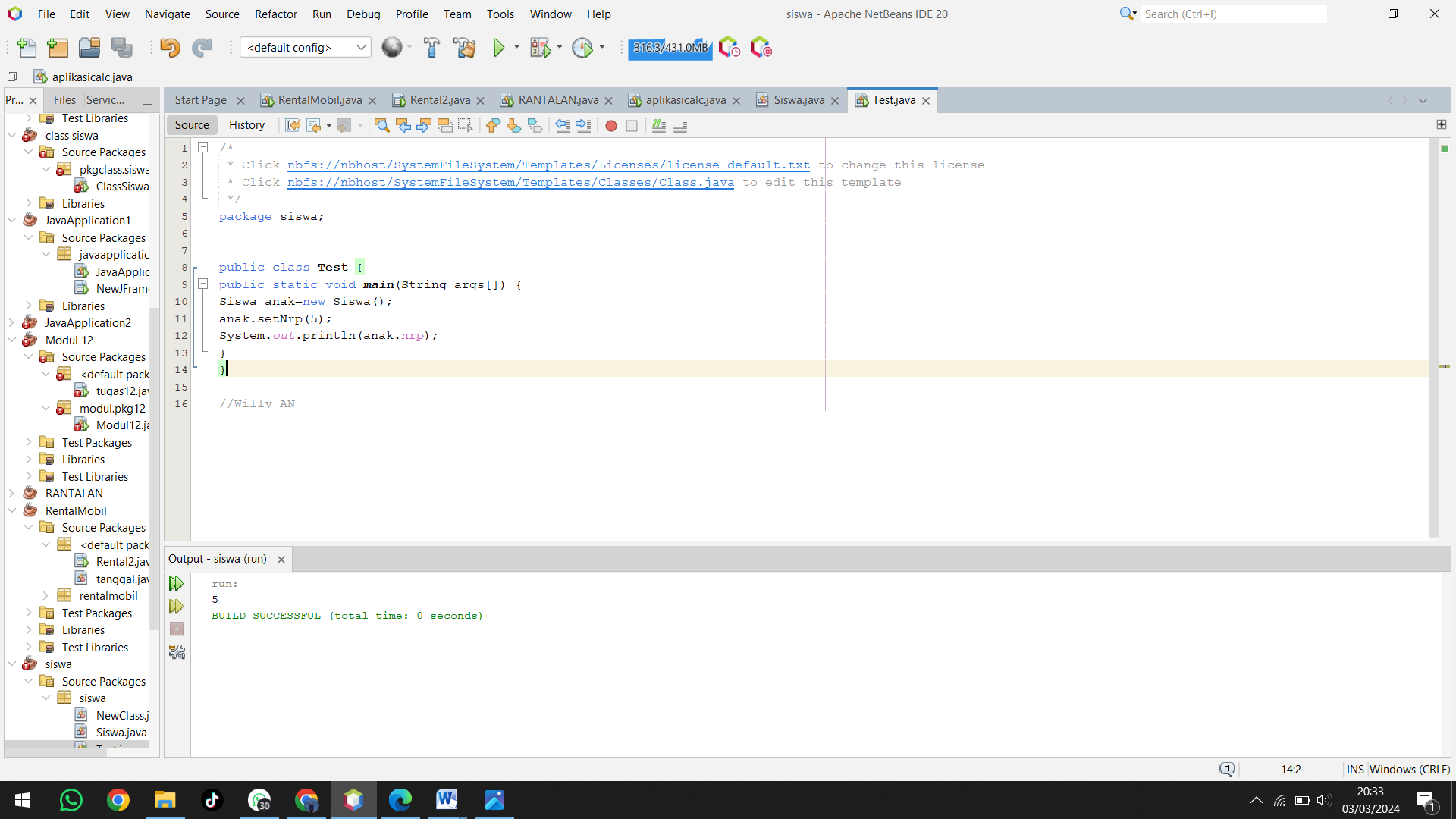
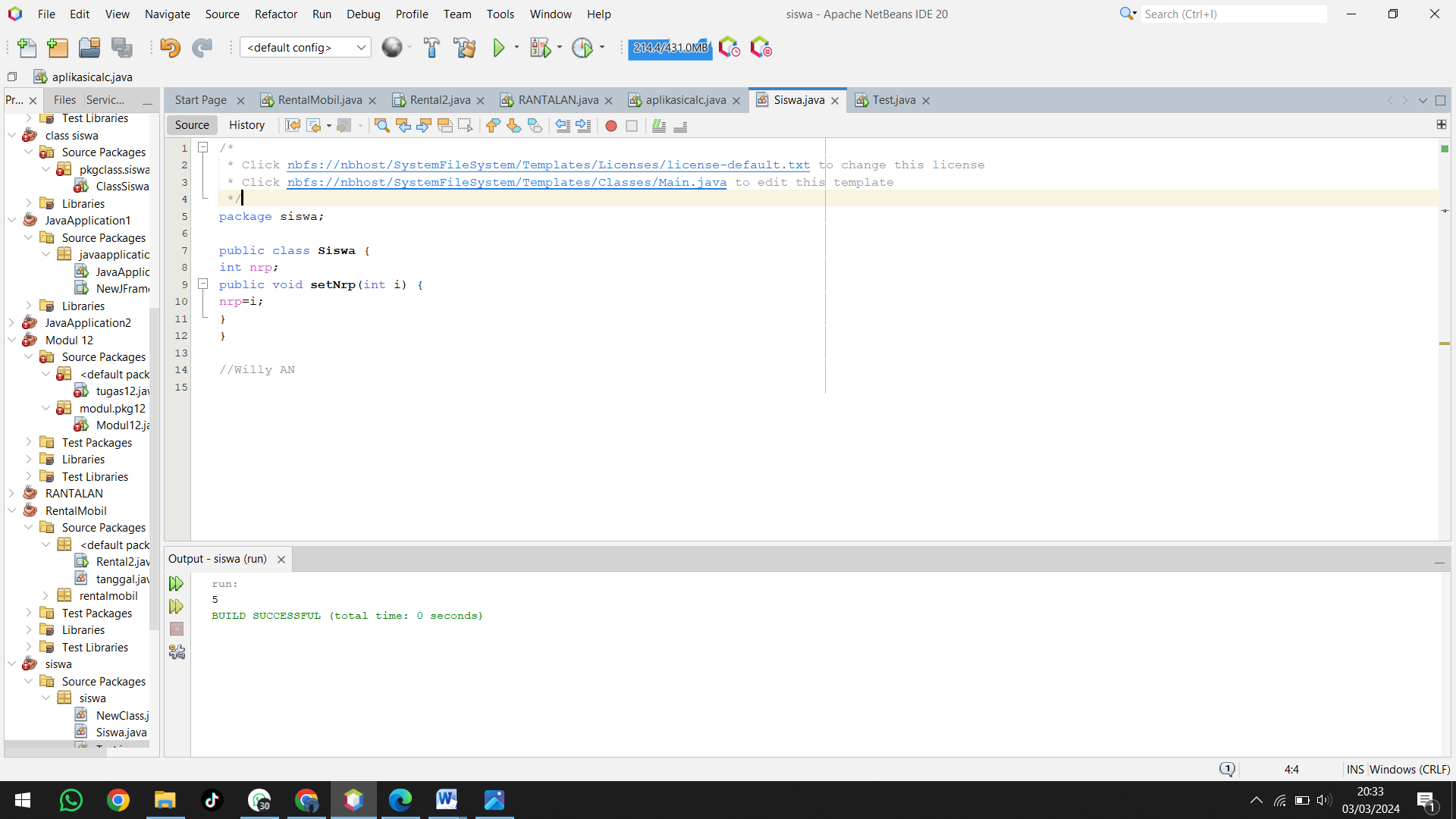
**- Membuat objek baru dari kelas namasiswa dengan nama anak.**

**- Mengatur nilai atribut nrp pada objek anak menjadi 5 dengan menggunakan anak.nrp = 5.**

**- Mencetak nilai atribut nrp dari objek anak ke konsol menggunakan System.out.println.**

**Jadi, program ini menciptakan objek dari kelas namasiswa, mengatur nilai atribut nrp, dan kemudian mencetak nilai tersebut ke konsol. Pada contoh ini, nilai yang dicetak adalah 5.**

**Nomor 2**



**public class Siswa {**

**int nrp;**

**public void setNrp(int i) {**

**nrp=i;**

**}**

**}**

**public class Test {**

**public static void main(String args[]) {**

**Siswa anak=new Siswa();**

**anak.setNrp(5);**

**System.out.println(anak.nrp);**

**}**

**}**

**Kelas Siswa: Ini adalah kelas yang mewakili entitas siswa.**

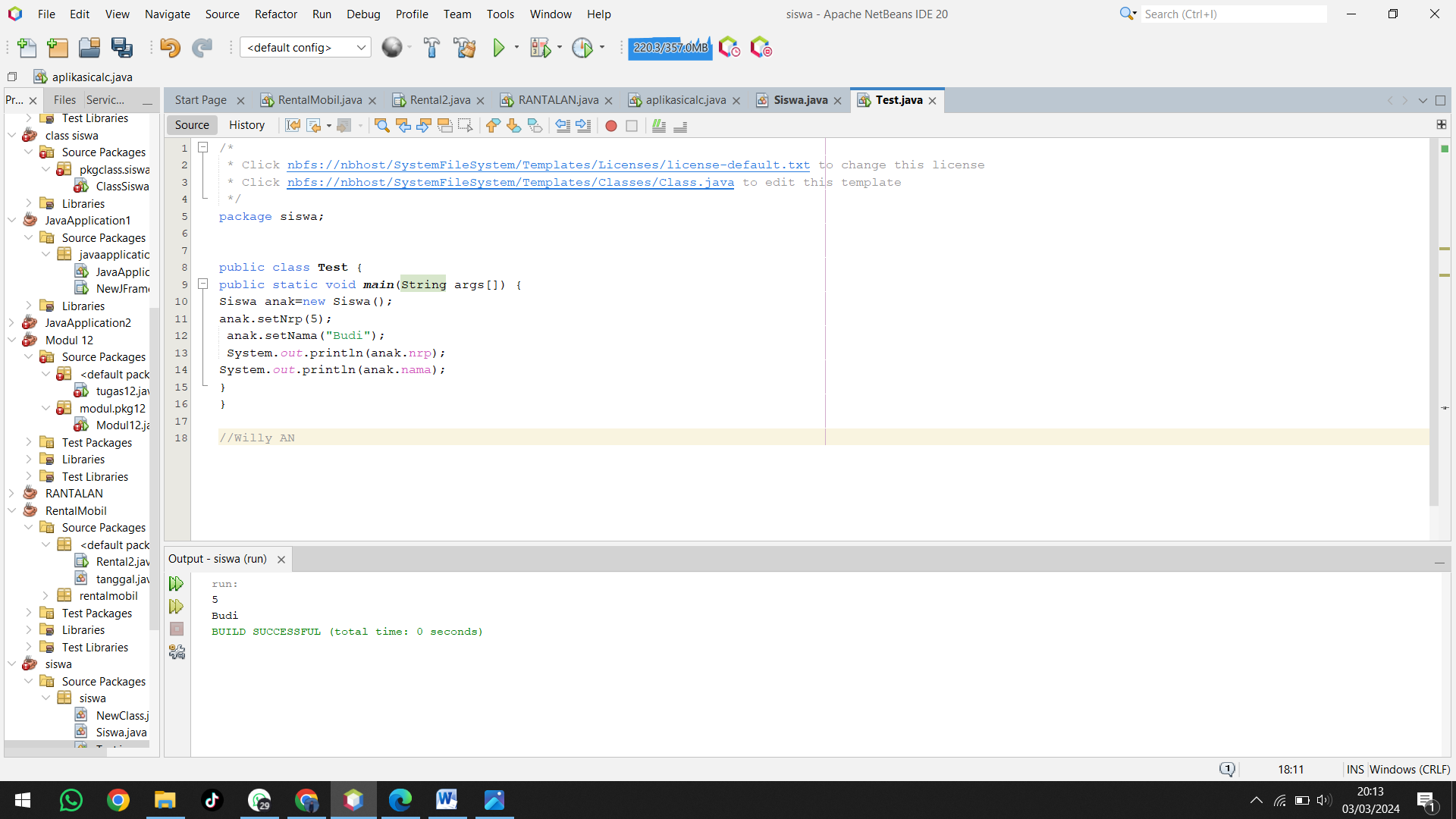
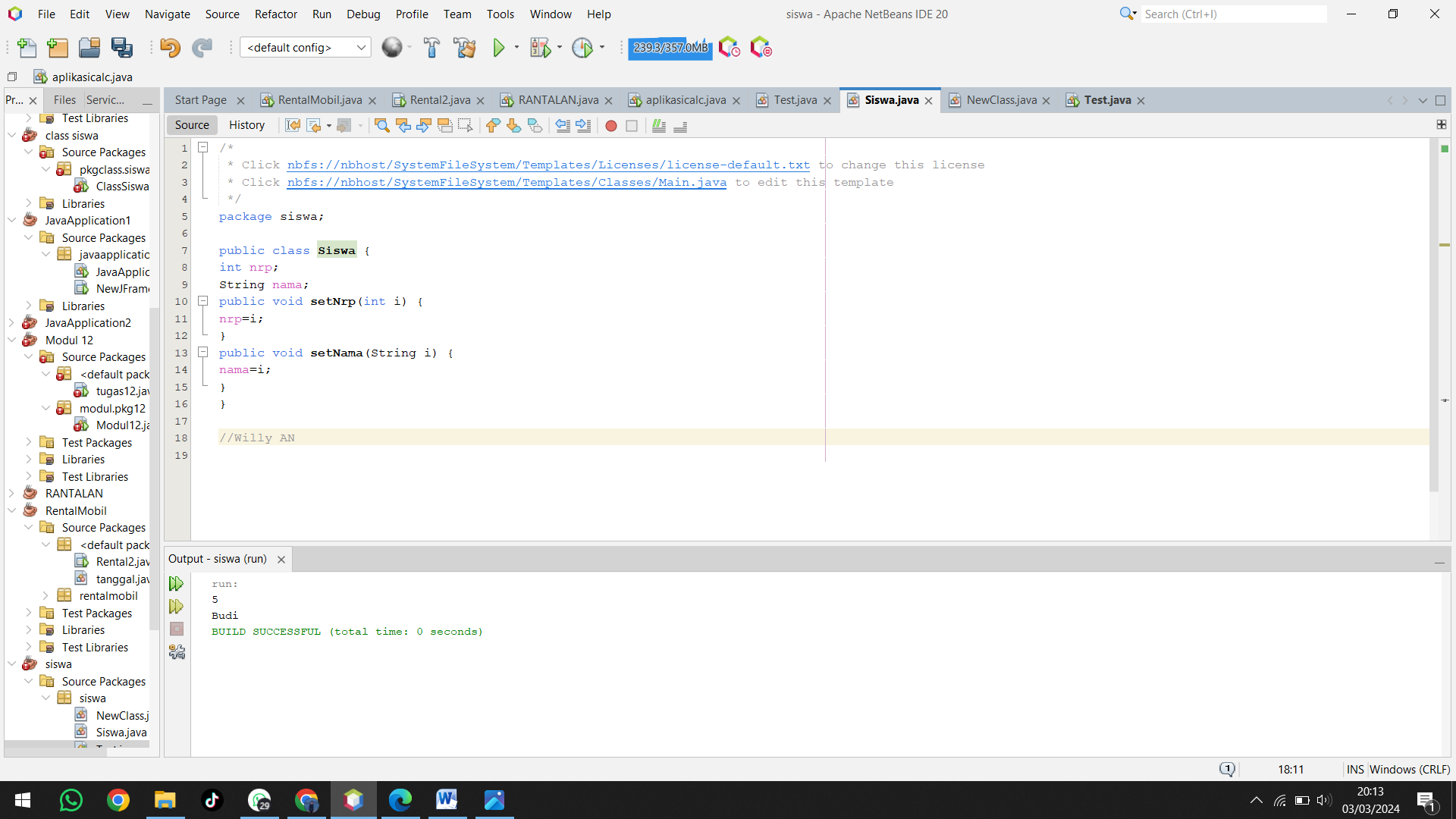
* **Variabel nrp: Ini adalah variabel instance yang mewakili Nomor Registrasi Siswa (NRP).**
* **Metode setNrp(int i): Ini adalah metode untuk menetapkan nilai nrp dengan nilai yang diberikan sebagai argumen i. Dalam hal ini, metode ini mengatur nilai nrp menjadi nilai i.**

**Kelas Siswa: Ini adalah kelas yang mewakili entitas siswa.**

* **Variabel nrp: Ini adalah variabel instance yang mewakili Nomor Registrasi Siswa (NRP).**
* **Metode setNrp(int i): Ini adalah metode untuk menetapkan nilai nrp dengan nilai yang diberikan sebagai argumen i. Dalam kasus ini, metode ini mengatur nilai nrp menjadi nilai i.**

**Jadi, program ini menciptakan objek mahasiswa, mengatur nomor registrasinya, dan kemudian mencetak nomor registrasi tersebut ke konsol. Pada contoh ini, nomor registrasi yang dicetak adalah 5.**

**Nomor 3**



**package siswa;**

**public class Siswa {**

**int nrp;**

**String nama;**

**public void setNrp(int i) {**

**nrp=i;**

**}**

**public void setNama(String i) {**

**nama=i;**

**}**

**}**

**public class Test {**

**public static void main(String args[]) {**

**Siswa anak=new Siswa();**

**anak.setNrp(5);**

**anak.setNama("Budi");**

**System.out.println(anak.nrp);**

**System.out.println(anak.nama);**

**}**

**}**

**Penjelasan: fungsi dari kodingan di atas:**

1. **Kelas Siswa:**

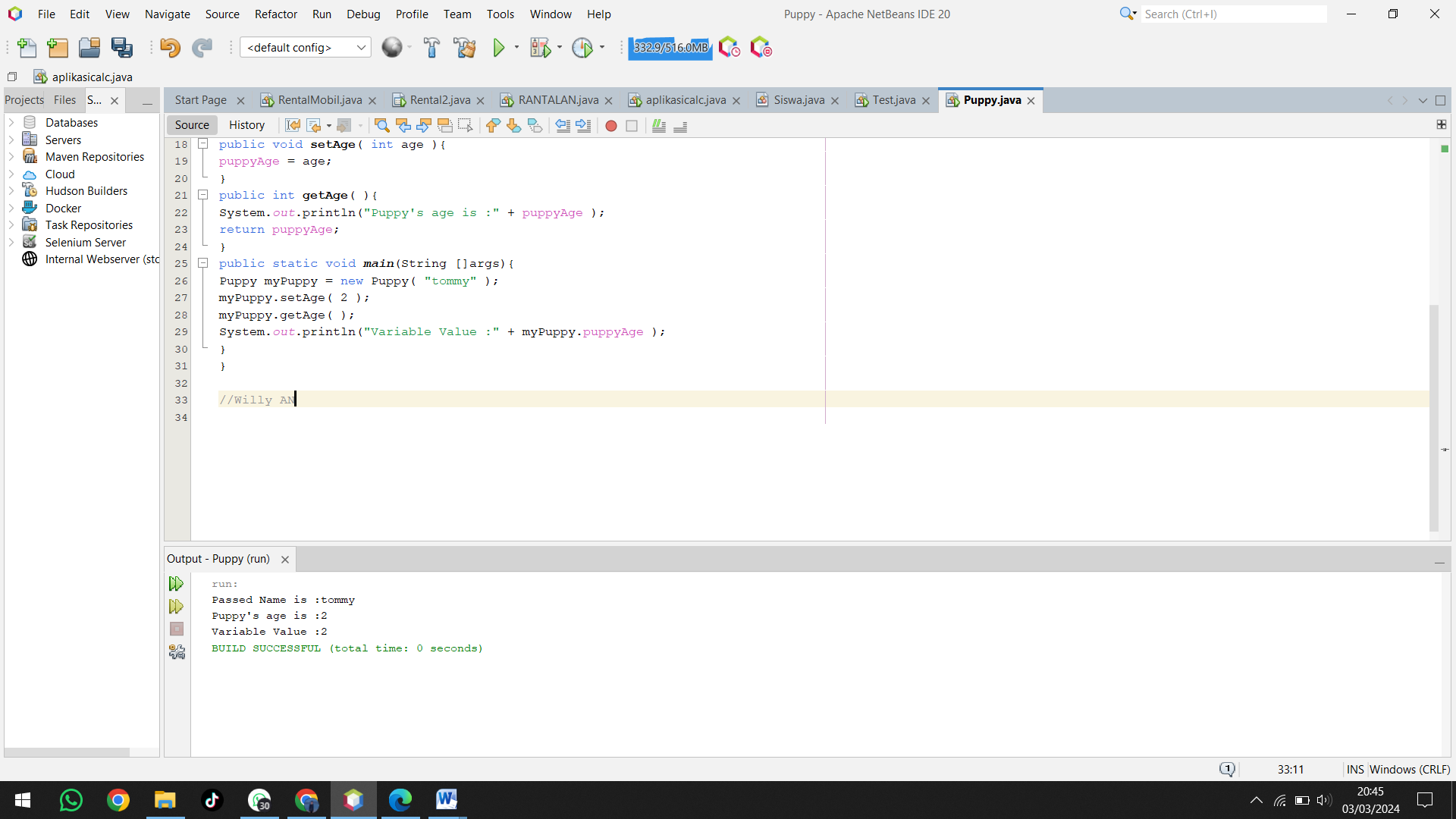
* **int nrp: Variabel untuk menyimpan nomor registrasi mahasiswa.**
* **String nama: Variabel untuk menyimpan nama mahasiswa.**
* **setNrp(int i): Metode untuk mengatur nilai nrp.**
* **setNama(String i): Metode untuk mengatur nilai nama.**

1. **Kelas Test:**

* **main(String args[]): Metode utama sebagai titik masuk eksekusi program.**
* **Siswa anak = new Siswa(): Membuat objek dari kelas Siswa dengan nama anak.**
* **anak.setNrp(5): Menetapkan nilai nrp objek anak menjadi 5.**
* **anak.setNama("Budi"): Menetapkan nilai nama objek anak menjadi "Budi".**
* **System.out.println(anak.nrp): Mencetak nilai nrp dari objek anak.**
* **System.out.println(anak.nama): Mencetak nilai nama dari objek anak.**

**Jadi, program ini menciptakan objek mahasiswa, mengatur nilai nrp dan nama melalui metode setter, dan kemudian mencetak nilai nrp dan nama tersebut. Dengan menggunakan kelas dan objek, ini menunjukkan konsep dasar pemrograman berorientasi objek di Java.**

**Nomor 4**



**class Puppy {**

**private int puppyAge;**

**public Puppy(String name){**

**// This constructor has one parameter, name.**

**System.out.println("Passed Name is :" + name );**

**}**

**public void setAge( int age ){**

**puppyAge = age;**

**}**

**public int getAge( ){**

**System.out.println("Puppy's age is :" + puppyAge );**

**return puppyAge;**

**}**

**public static void main(String []args){**

**Puppy myPuppy = new Puppy( "tommy" );**

**myPuppy.setAge( 2 );**

**myPuppy.getAge( );**

**System.out.println("Variable Value :" + myPuppy.puppyAge );**

**}**

**}**

**Penjelasan: Program di atas adalah implementasi kelas Puppy dalam bahasa pemrograman Java, yang memiliki atribut puppyAge dan beberapa metode untuk mengatur dan mendapatkan nilai usia seekor anak anjing. Berikut adalah penjelasan fungsi programnya:**

1. **Deklarasi Kelas Puppy:**

* **Kelas ini memiliki atribut privat (private int puppyAge) untuk menyimpan usia anak anjing.**

1. **Konstruktor Puppy(String name):**

* **Konstruktor menerima parameter nama dan mencetak pesan yang menyatakan nama yang dilewatkan.**

1. **Metode setAge(int age):**

* **Metode ini digunakan untuk mengatur nilai atribut puppyAge dengan nilai usia yang diberikan sebagai argumen.**

1. **Metode getAge():**

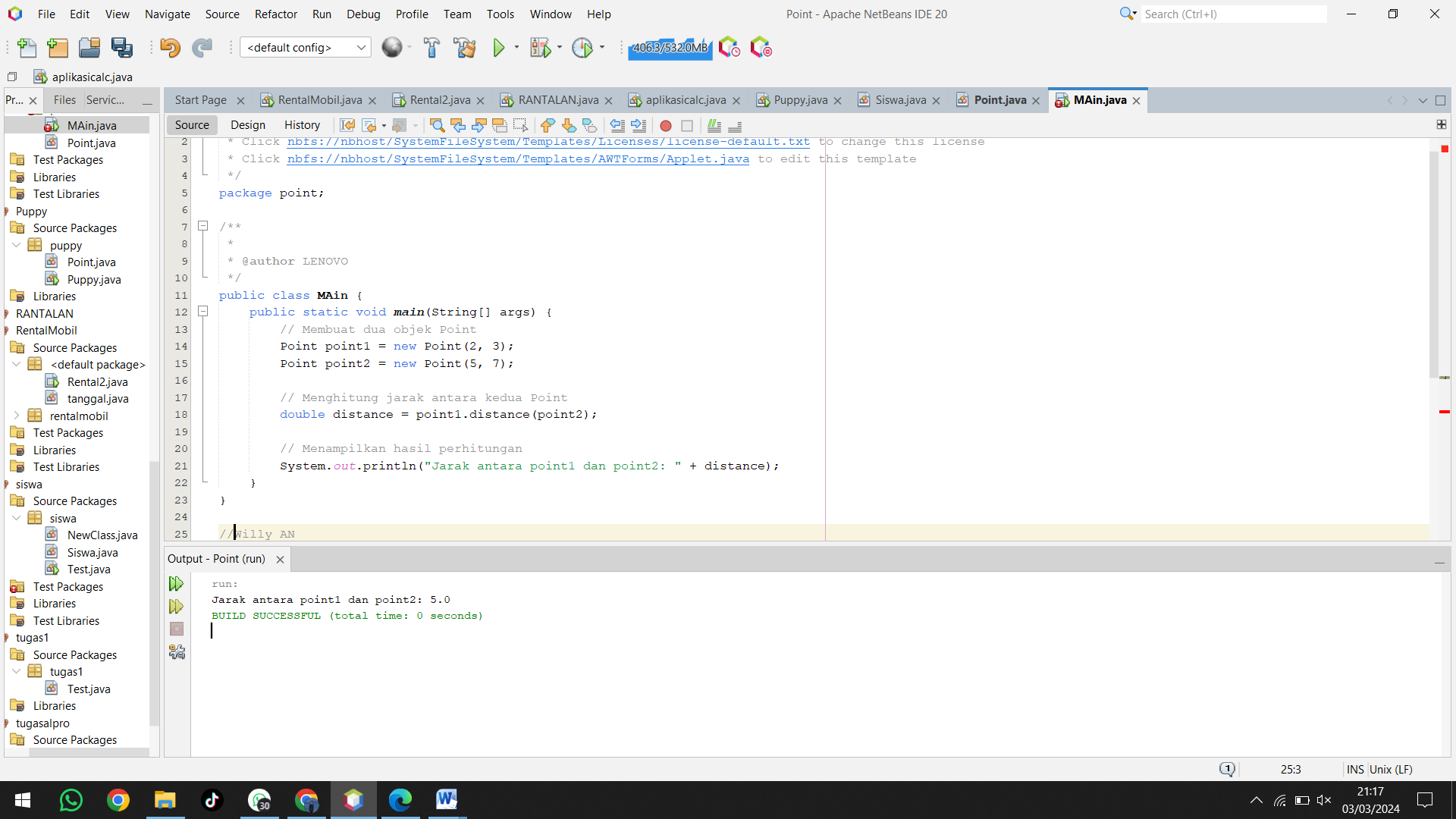
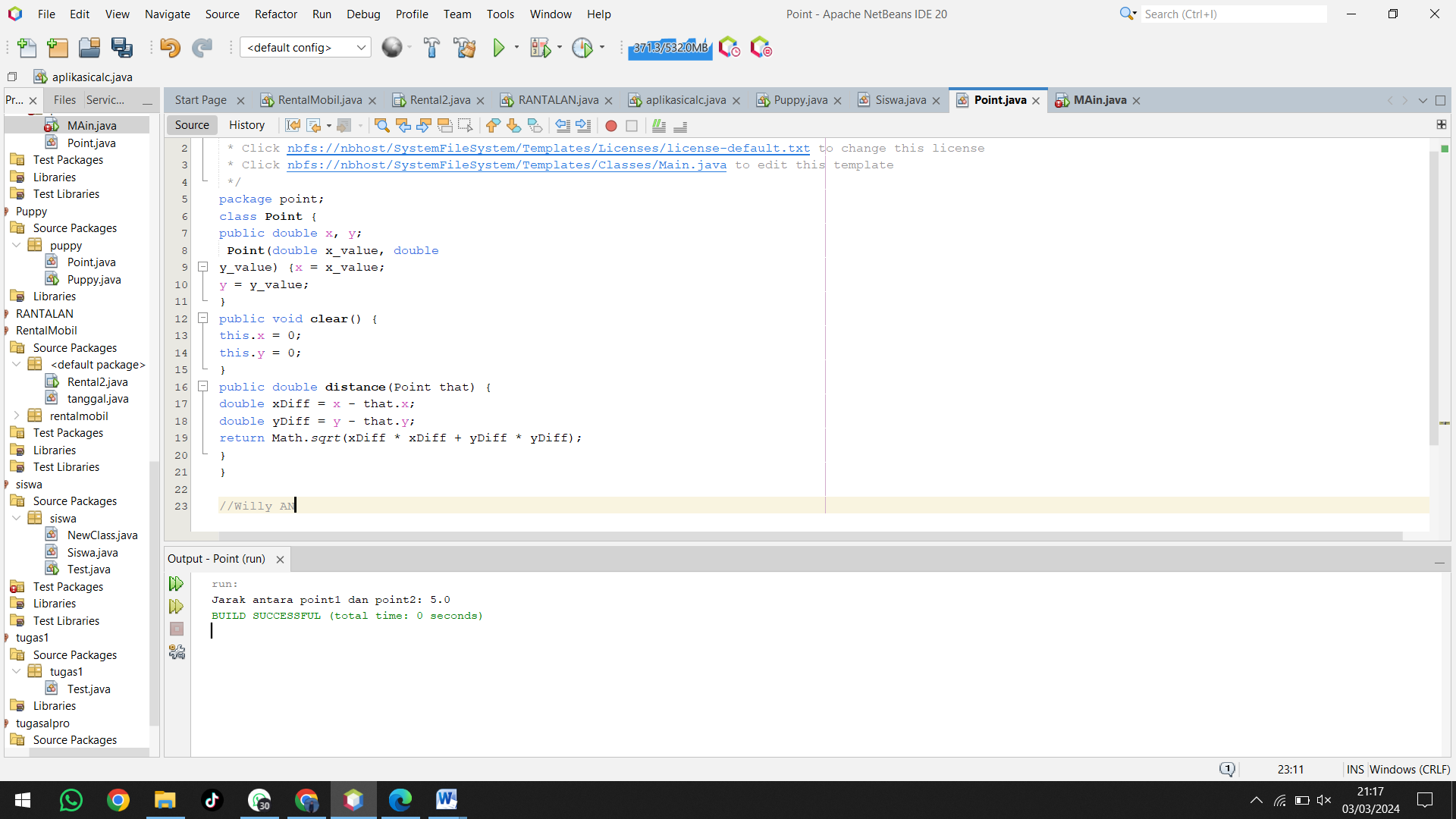
* **Metode ini mencetak usia anjing ke konsol dan mengembalikan nilai usia tersebut.**

1. **Metode main(String[] args):**

* **Metode utama program yang membuat objek myPuppy dari kelas Puppy dengan nama "tommy".**
* **Mengatur usia anak anjing menggunakan metode setAge(2).**
* **Mendapatkan dan mencetak usia anak anjing ke konsol dengan metode getAge().**
* **Mencetak nilai atribut puppyAge secara langsung ke konsol.**

**Jadi, program ini menciptakan objek Puppy, mengatur dan mendapatkan nilai usia anak anjing, dan mencetak informasi terkait ke konsol. Pada contoh ini, nama anjing dicetak saat objek dibuat, usia diatur dan dicetak, serta nilai atribut usia diakses dan dicetak langsung.**

**Nomor 5**



**package point;**

**class Point {**

**public double x, y;**

**Point(double x\_value, double**

**y\_value) {x = x\_value;**

**y = y\_value;**

**}**

**public void clear() {**

**this.x = 0;**

**this.y = 0;**

**}**

**public double distance(Point that) {**

**double xDiff = x - that.x;**

**double yDiff = y - that.y;**

**return Math.sqrt(xDiff \* xDiff + yDiff \* yDiff);**

**}**

**}**

**public class MAin {**

**public static void main(String[] args) {**

**// Membuat dua objek Point**

**Point point1 = new Point(2, 3);**

**Point point2 = new Point(5, 7);**

**// Menghitung jarak antara kedua Point**

**double distance = point1.distance(point2);**

**// Menampilkan hasil perhitungan**

**System.out.println("Jarak antara point1 dan point2: " + distance);**

**}**

**}**

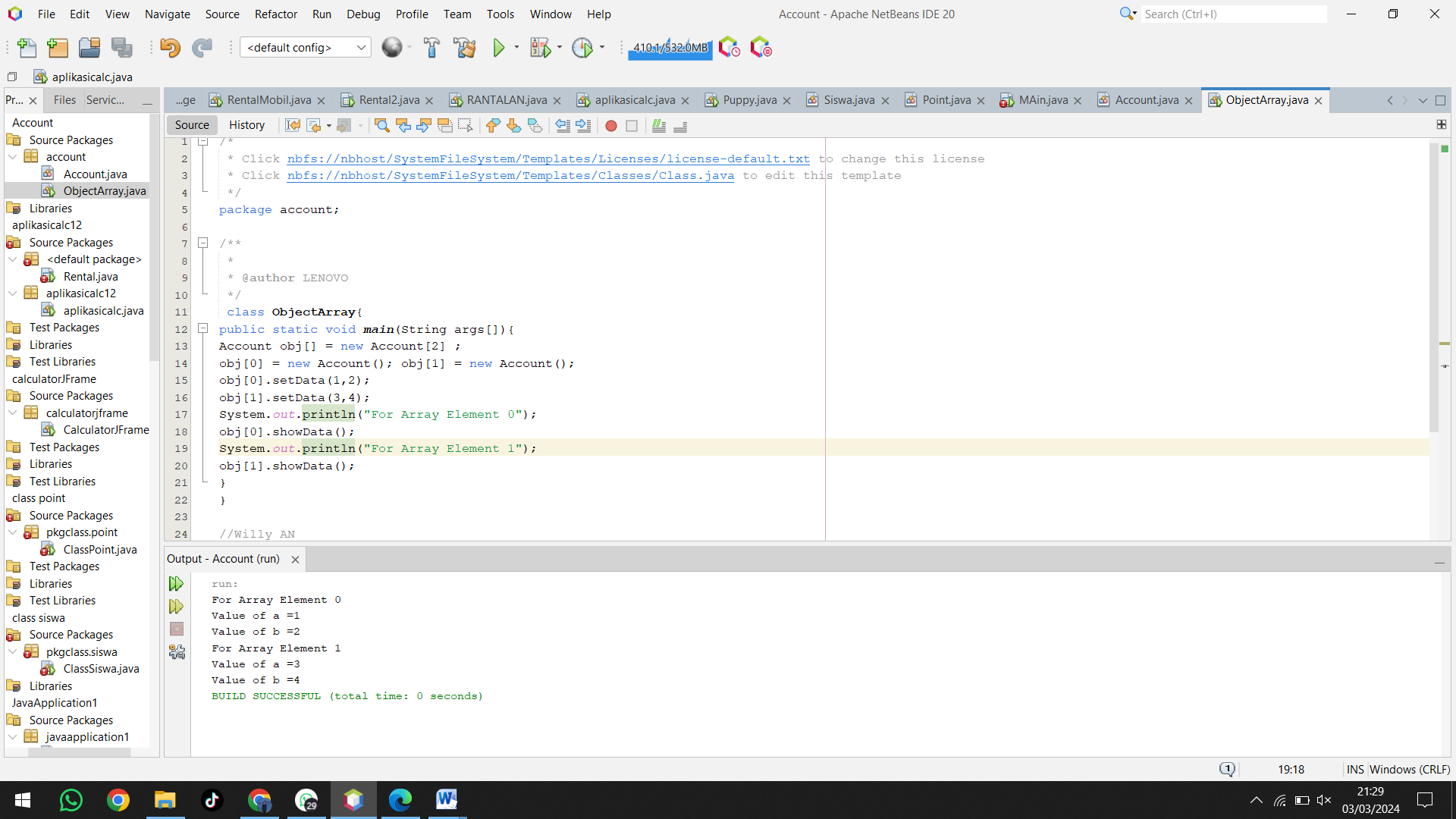
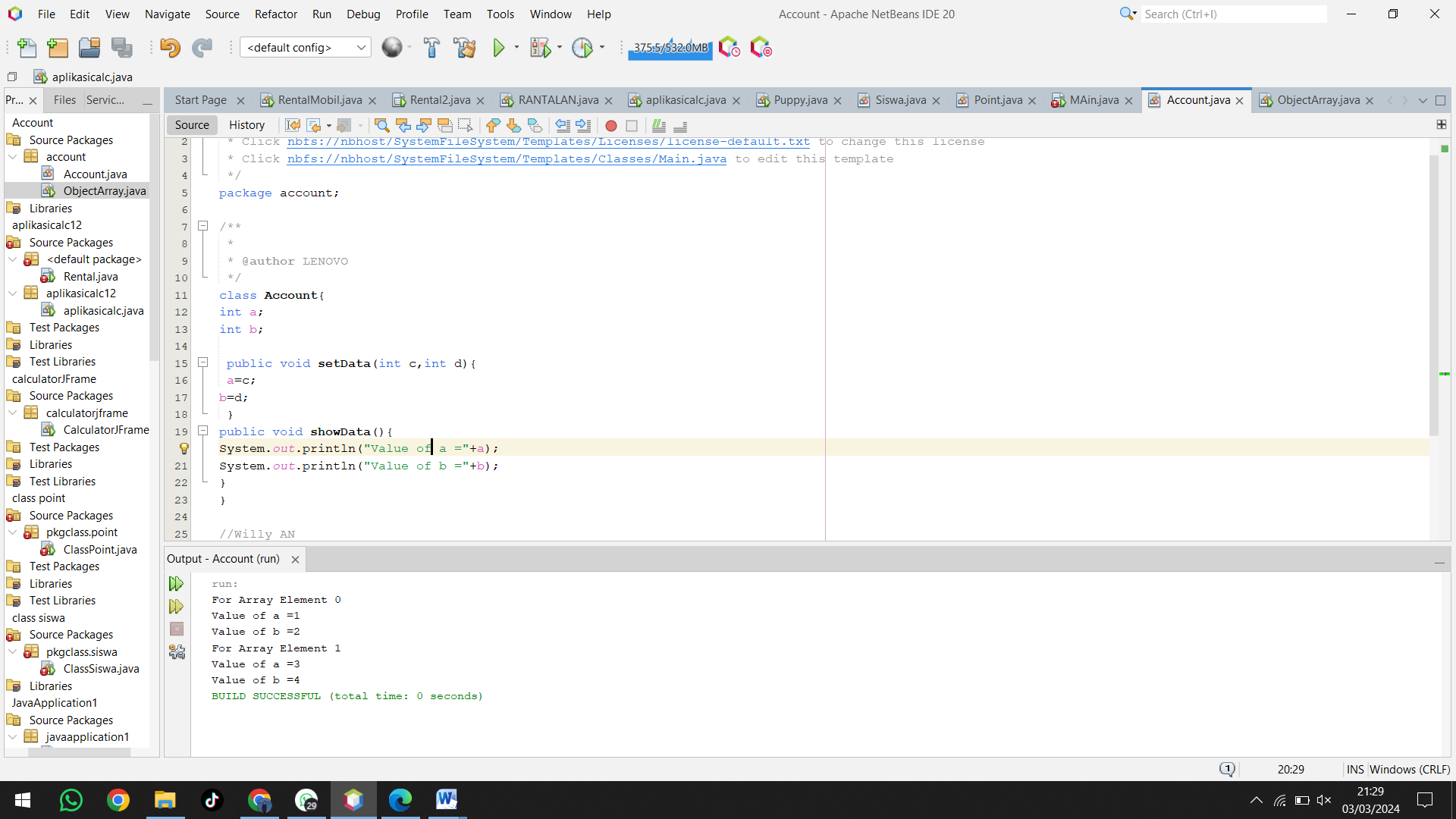
**Kelas Point: Ini adalah kelas yang merepresentasikan titik dalam sistem koordinat kartesian.**

* **Variabel x dan y: Ini adalah variabel instance yang menyimpan koordinat titik.**
* **Konstruktor Point(double x\_value, double y\_value): Ini adalah konstruktor kelas yang menginisialisasi nilai x dan y saat objek Point dibuat.**
* **Metode clear(): Ini adalah metode yang digunakan untuk mengatur kedua nilai x dan y menjadi 0.**
* **Metode distance(Point that): Ini adalah metode yang menghitung jarak antara titik saat ini dan titik yang diberikan sebagai argumen that. Metode ini menghitung jarak menggunakan rumus jarak Euclidean (jarak antara dua titik dalam ruang dua dimensi) dan mengembalikan nilai jarak tersebut.**

**Kelas MAin: Ini adalah kelas utama yang memiliki metode main sebagai titik masuk utama program.**

* **Metode main(String[] args): Ini adalah metode yang dieksekusi pertama kali saat program dijalankan.**
* **Membuat dua objek Point, yaitu point1 dengan koordinat (2, 3) dan point2 dengan koordinat (5, 7).**
* **Menghitung jarak antara dua titik menggunakan metode distance(Point that) dari objek point1 dengan point2 sebagai argumen.**
* **Menampilkan hasil perhitungan jarak ke layar menggunakan System.out.println().**

**Nomor 6**



**class Account{**

**int a;**

**int b;**

**public void setData(int c,int d){**

**a=c;**

**b=d;**

**}**

**public void showData(){**

**System.out.println("Value of a ="+a);**

**System.out.println("Value of b ="+b);**

**}**

**}**

**class ObjectArray{**

**public static void main(String args[]){**

**Account obj[] = new Account[2] ;**

**obj[0] = new Account(); obj[1] = new Account();**

**obj[0].setData(1,2);**

**obj[1].setData(3,4);**

**System.out.println("For Array Element 0");**

**obj[0].showData();**

**System.out.println("For Array Element 1");**

**obj[1].showData();**

**}**

**}**

**Kelas Account: Ini adalah kelas yang memiliki dua variabel instance a dan b, yang mungkin mewakili data akun atau informasi lainnya.**

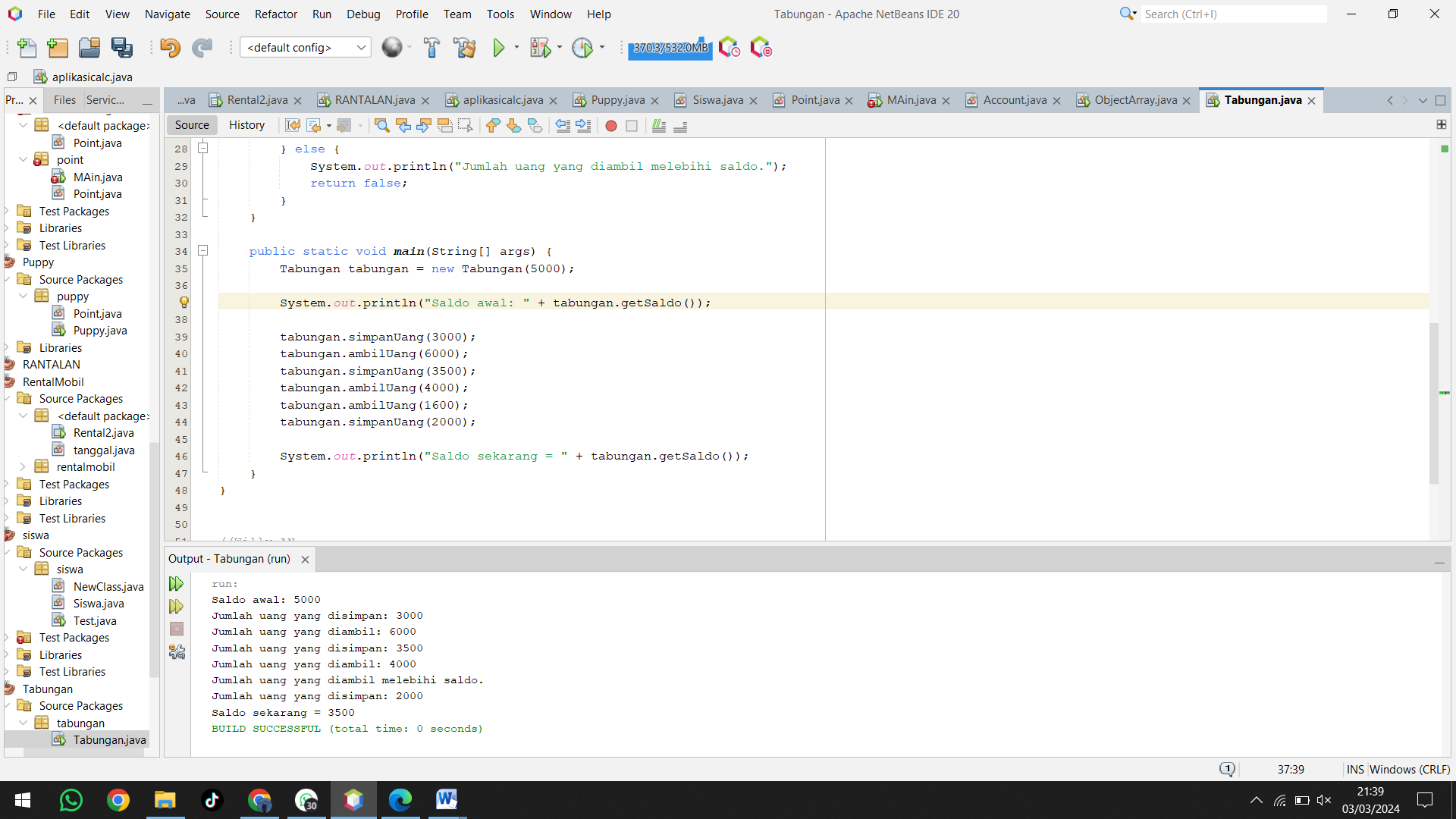
* **Metode setData(int c, int d): Ini adalah metode yang digunakan untuk mengatur nilai variabel instance a dan b dengan nilai yang diberikan sebagai argumen c dan d.**
* **Metode showData(): Ini adalah metode yang digunakan untuk menampilkan nilai dari variabel instance a dan b ke layar.**

**Kelas ObjectArray: Ini adalah kelas yang memiliki metode main sebagai titik masuk utama program.**

* **Metode main(String args[]): Ini adalah metode yang dieksekusi pertama kali saat program dijalankan.**
* **Membuat sebuah array obj yang terdiri dari dua elemen yang bertipe Account. Array ini akan menampung dua objek dari kelas Account.**
* **Membuat dua objek dari kelas Account menggunakan konstruktor default dan menyimpannya di dalam array obj.**
* **Mengatur data untuk masing-masing objek yang disimpan dalam array obj, yaitu objek pada indeks 0 dan 1, menggunakan metode setData(int c, int d).**
* **Menampilkan data untuk masing-masing objek yang disimpan dalam array obj menggunakan metode showData(). Ini dilakukan dengan mengakses objek pada indeks 0 dan 1 dari array obj.**

**E. Latihan**

**Latihan 1**



**Penjelasan : Program di atas adalah implementasi sederhana dari kelas Tabungan dalam bahasa pemrograman Java. Berikut adalah penjelasan fungsi programnya:**

1. **Deklarasi Kelas Tabungan:**

* **Kelas ini memiliki atribut privat (private int saldo) untuk menyimpan saldo tabungan.**

1. **Konstruktor Tabungan(int initSaldo):**

* **Konstruktor ini digunakan untuk membuat objek Tabungan baru dengan saldo awal yang diberikan sebagai argumen saat objek dibuat.**

1. **Metode getSaldo():**

* **Metode ini mengembalikan nilai saldo tabungan.**

1. **Metode simpanUang(int jumlah):**

* **Metode ini digunakan untuk menambahkan jumlah uang ke saldo tabungan.**

1. **Metode ambilUang(int jumlah):**

* **Metode ini memeriksa apakah saldo mencukupi untuk pengambilan uang sejumlah tertentu.**
* **Jika mencukupi, saldo dikurangkan dan metode mengembalikan true.**
* **Jika tidak mencukupi, metode mengembalikan false.**

1. **Metode transfer(Tabungan tab, int jumlah):**

* **Metode ini memeriksa apakah saldo mencukupi untuk transfer sejumlah tertentu.**
* **Jika mencukupi, saldo objek saat ini dikurangkan dan saldo objek tab ditambahkan.**
* **Metode mengembalikan true untuk transfer berhasil.**
* **Jika saldo tidak mencukupi, metode mengembalikan false.**

1. **Metode main(String[] args):**

* **Metode utama program yang membuat dua objek Tabungan (tabungan1 dan tabungan2) dengan saldo awal masing-masing.**
* **Menampilkan saldo awal kedua tabungan.**
* **Menambahkan uang ke tabungan1, menampilkan saldo baru.**
* **Mencoba transfer dari tabungan1 ke tabungan2, menampilkan hasil dan saldo kedua tabungan setelah transfer.**

**Jadi, program ini menyimulasikan penggunaan objek Tabungan dengan menyimpan, mengambil uang, dan melakukan transfer saldo antar dua tabungan.**

**Latihan 2**

**public class Tabungan {**

**private int saldo;**

**// Konstruktor untuk menginisialisasi saldo awal**

**public Tabungan(int initsaldo) {**

**saldo = initsaldo;**

**}**

**// Metode untuk mendapatkan saldo**

**public int getSaldo() {**

**return saldo;**

**}**

**// Metode untuk menyimpan uang ke saldo**

**public void simpanUang(int jumlah) {**

**saldo += jumlah;**

**System.out.println("Jumlah uang yang disimpan: " + jumlah);**

**}**

**// Metode untuk mengambil uang dari saldo**

**public boolean ambilUang(int jumlah) {**

**if (jumlah <= saldo) {**

**saldo -= jumlah;**

**System.out.println("Jumlah uang yang diambil: " + jumlah);**

**return true;**

**} else {**

**System.out.println("Jumlah uang yang diambil melebihi saldo.");**

**return false;**

**}**

**}**

**// Metode untuk mentransfer sejumlah uang ke tabungan lain**

**public boolean transfer(Tabungan tab, int jumlah) {**

**if (ambilUang(jumlah)) {**

**tab.simpanUang(jumlah);**

**return true;**

**} else {**

**System.out.println("Transfer gagal: Saldo tidak mencukupi.");**

**return false;**

**}**

**}**

**public static void main(String[] args) {**

**// Membuat objek tabungan**

**Tabungan tabungan1 = new Tabungan(5000);**

**Tabungan tabungan2 = new Tabungan(0); // Saldo awal tabungan kedua 0**

**// Menampilkan saldo awal tabungan**

**System.out.println("Saldo awal tabungan 1: " + tabungan1.getSaldo());**

**System.out.println("Saldo awal tabungan 2: " + tabungan2.getSaldo());**

**// Menyimpan uang, mengambil uang, dan melakukan transfer**

**tabungan1.simpanUang(3000);**

**tabungan1.ambilUang(6000);**

**tabungan1.simpanUang(3500);**

**tabungan1.ambilUang(4000);**

**tabungan1.ambilUang(1600);**

**tabungan1.simpanUang(2000);**

**// Menampilkan saldo tabungan setelah operasi**

**System.out.println("Saldo sekarang tabungan 1: " + tabungan1.getSaldo());**

**// Melakukan transfer antar tabungan**

**tabungan1.transfer(tabungan2, 1500);**

**tabungan1.transfer(tabungan2, 5000);**

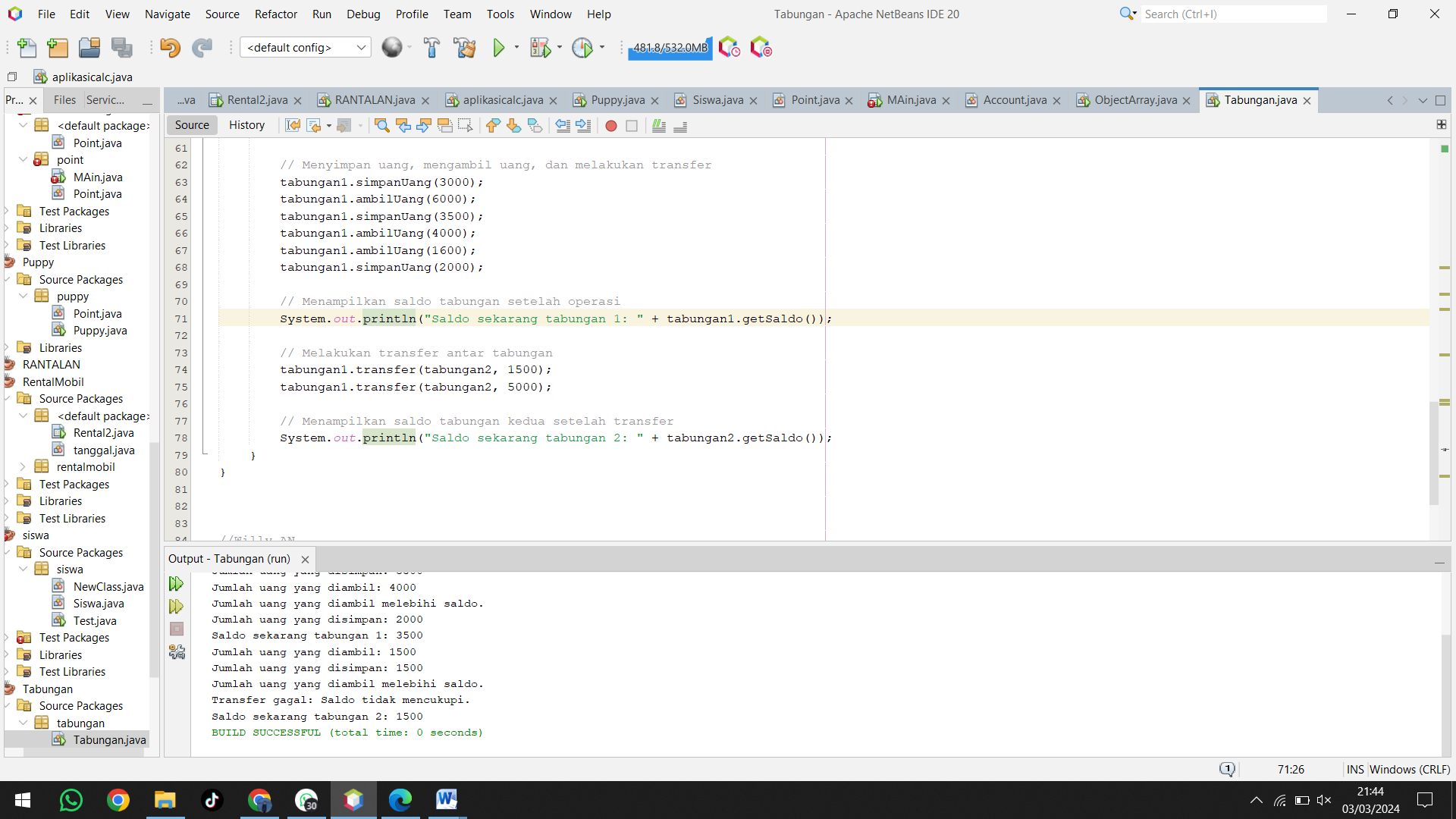
**// Menampilkan saldo tabungan kedua setelah transfer**

**System.out.println("Saldo sekarang tabungan 2: " + tabungan2.getSaldo());**

**}**

**}**

**Output :**



**Kelas Tabungan: Ini adalah kelas yang mengelola saldo tabungan.**

* **Atribut saldo: Menyimpan saldo tabungan.**
* **Konstruktor Tabungan(int initsaldo): Digunakan untuk menginisialisasi saldo awal tabungan.**
* **Metode getSaldo(): Mengembalikan nilai saldo tabungan.**
* **Metode simpanUang(int jumlah): Menambahkan jumlah uang ke saldo tabungan.**
* **Metode ambilUang(int jumlah): Mengurangi jumlah uang dari saldo tabungan jika saldo mencukupi.**
* **Metode transfer(Tabungan tab, int jumlah): Mentransfer sejumlah uang dari tabungan saat ini ke tabungan lain jika saldo mencukupi.**