**UJIAN TENGAH PRAKTIKUM**

**ALGORITMA STRUKTUR DATA**

**“KESEHATAN: PROGRAM MANAJEMEN DATA PASIEN RUMAH SAKIT”**



Nama : Muhammad Azka Raki

NIM : 2311016110005

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**

**UNIVERSITAS LAMBUNG MANGKURAT**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**PROGRAM STUDI S1 ILMU KOMPUTER**

**BANJARBARU**

**2024**

1. **Tujuan dan Urgensi Memilih Kasus**

Tujuan saya untuk memilih kasus ini adalah sebagai rancangan program sederhana untuk manajemen pasien di rumah sakit. Dengan banyaknya pasien yang mengunjungi atau yang pasien yang rawat inap dalam suatu rumah sakit, program sederhana ini diharapkan bisa membantu dalam memanajemen pasien-pasien tersebut. Tujuan lainnya selain manajemen pasien adalah pengguna dapat dengan mudah mencari pasien dan mensortir pasien-pasien berdasarkan kategori yang tersedia. Urgensi saya dalam memilih kasus ini adalah keperluan efisiensi dan keakuratan dalam mengelola data pasien dan digitalisasi pengelolaan data pasien di rumah sakit.

1. **Source Code**
2. Main.java

|  |
| --- |
| package utp;  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scan = new Scanner(System.in);  ArrayPasien arrayPasien = new ArrayPasien();  int pilihan;  String namaLengkap;  do {  menu();  System.out.print("Masukkan Nomor Pilihan : ");  pilihan = scan.nextInt();  switch (pilihan) {  case 1:  Pasien pasienBaru = new Pasien();  pasienBaru.isiDataPasien();  if (arrayPasien.tambahPasien(pasienBaru)) {  System.out.println("Pasien atas nama " + pasienBaru.getNamaLengkap() + " berhasil ditambahkan");  } else {  System.out.println("Array pasien telah penuh. Coba hapus pasien kemudian tambah lagi");  }  break;  case 2:  System.out.print("Masukkan Nama Lengkap : ");  namaLengkap = scan.nextLine();  namaLengkap = scan.nextLine();  if (arrayPasien.hapusPasien(namaLengkap)) {  System.out.println("Pasien atas nama " + namaLengkap + " berhasil dihapus");  } else {  System.out.println("Pasien atas nama " + namaLengkap + " tidak ditemukan");  }  break;  case 3:  System.out.print("Masukkan Nama Lengkap : ");  namaLengkap = scan.nextLine();  namaLengkap = scan.nextLine();  int index = arrayPasien.cariIndexPasien(namaLengkap);  if (index != -1) {  Pasien[] arrayDataPasien = arrayPasien.getArrayPasien();  Pasien dataPasien = arrayDataPasien[index];  System.out.println("Pasien atas nama " + dataPasien.getNamaLengkap() + " berhasil ditemukan.");  arrayPasien.printDataPasien(dataPasien);  } else {  System.out.println("Pasian atas nama " + namaLengkap + " tidak ditemukan");  }  break;  case 4:  if (arrayPasien.getJumlahPasien() == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  break;  }  arrayPasien.quickSortBerdasarkanNamaLengkap(0, arrayPasien.getJumlahPasien() - 1);  System.out.println("Array Pasien Setelah Diurutkan Berdasarkan Nama Lengkap : ");  arrayPasien.printSemuaDataPasien(0);  break;  case 5:  if (arrayPasien.getJumlahPasien() == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  break;  }  arrayPasien.quickSortBerdasarkanBeratBadan(0, arrayPasien.getJumlahPasien() - 1);  System.out.println("Array Pasien Setelah Diurutkan Berdasarkan Berat Badan : ");  arrayPasien.printSemuaDataPasien(0);  break;  case 6:  if (arrayPasien.getJumlahPasien() == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  break;  }  arrayPasien.quickSortBerdasarkanTinggiBadan(0, arrayPasien.getJumlahPasien() - 1);  System.out.println("Array Pasien Setelah Diurutkan Berdasarkan Tinggi Badan : ");  arrayPasien.printSemuaDataPasien(0);  break;  case 7:  arrayPasien.printSemuaDataPasien(0);  break;  case 8:  break;  default:  System.out.println("Input INVALID. ERROR");  }  System.out.println("-------------------------------------------------");  } while (pilihan != 8);  }  static void menu() {  System.out.println("");  System.out.println("-> Program Manajemen Data Pasien Rumah Sakit <-");  System.out.println("-------------------------------------------------");  System.out.println("Menu :");  System.out.println("1. Tambah Pasien");  System.out.println("2. Hapus Pasien");  System.out.println("3. Cari Pasien Berdasarkan Nama Lengkap");  System.out.println("4. Sorting Data-data Pasien Berdasarkan Nama");  System.out.println("5. Sorting Data-data Pasien Berdasarkan Berat Badan");  System.out.println("6. Sorting Data-data Pasien Berdasarkan Tinggi Badan");  System.out.println("7. Tampilkan Semua Data Pasien");  System.out.println("8. Keluar Dari Program");  System.out.println("-------------------------------------------------");  }  } |

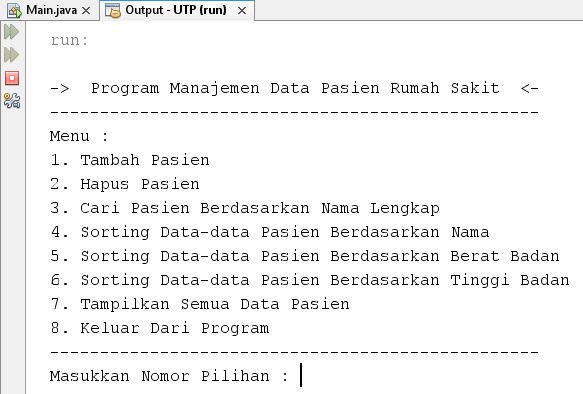
1. ArrayPasien.java

|  |
| --- |
| package utp;  public class ArrayPasien {  static int length = 15;  private Pasien[] arrayPasien = new Pasien[length];  private int jumlahPasien = 0;  public int getJumlahPasien() {  return jumlahPasien;  }  public Pasien[] getArrayPasien() {  return arrayPasien;  }  boolean tambahPasien(Pasien pasien) {  if (jumlahPasien < length) {  arrayPasien[jumlahPasien] = pasien;  jumlahPasien++;  return true;  } else {  return false;  }  }  boolean hapusPasien(String namaLengkap) {  if (this.jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return false;  }  int index = cariIndexPasien(namaLengkap);  if (index == -1) {  return false;  }  jumlahPasien--;  hapusPasienRekursi(index);  return true;  }  void hapusPasienRekursi(int index) {  if (index == this.jumlahPasien) {  return;  }  arrayPasien[index] = arrayPasien[index + 1];  hapusPasienRekursi(++index);  }  int cariIndexPasien(String namaLengkap) {  if (jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return -1;  }  for (int i = 0; i < jumlahPasien; i++) {  Pasien pasien = arrayPasien[i];  if (pasien.getNamaLengkap().equalsIgnoreCase(namaLengkap)) {  return i;  }  }  return -1;  }  void quickSortBerdasarkanBeratBadan(int low, int high) {  if (jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return;  }  if (low < high) {  int pivotIndex = partitionBeratBadan(low, high);  quickSortBerdasarkanBeratBadan(low, pivotIndex - 1);  quickSortBerdasarkanBeratBadan(pivotIndex + 1, high);  }  }  private int partitionBeratBadan(int low, int high) {  double pivot = arrayPasien[high].getBeratBadan();  int i = low - 1;  i = rekurcionSwapDataBeratBadan(low, high, pivot, i);  Pasien temp = arrayPasien[i + 1];  arrayPasien[i + 1] = arrayPasien[high];  arrayPasien[high] = temp;  return i + 1;  }  private int rekurcionSwapDataBeratBadan(int j, int high, double pivot, int i) {  if (j >= high) {  return i;  }  double data = arrayPasien[j].getBeratBadan();  if (data <= pivot) {  i++;  Pasien temp = arrayPasien[i];  arrayPasien[i] = arrayPasien[j];  arrayPasien[j] = temp;  }  j += 1;  return rekurcionSwapDataBeratBadan(j, high, pivot, i);  }  void quickSortBerdasarkanTinggiBadan(int low, int high) {  if (jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return;  }  if (low < high) {  int pivotIndex = partitionTinggiBadan(low, high);  quickSortBerdasarkanTinggiBadan(low, pivotIndex - 1);  quickSortBerdasarkanTinggiBadan(pivotIndex + 1, high);  }  }  private int partitionTinggiBadan(int low, int high) {  double pivot = arrayPasien[high].getTinggiBadan();  int i = low - 1;  i = rekurcionSwapDataTinggiBadan(low, high, pivot, i);  Pasien temp = arrayPasien[i + 1];  arrayPasien[i + 1] = arrayPasien[high];  arrayPasien[high] = temp;  return i + 1;  }  private int rekurcionSwapDataTinggiBadan(int j, int high, double pivot, int i) {  if (j >= high) {  return i;  }  double data = arrayPasien[j].getTinggiBadan();  if (data <= pivot) {  i++;  Pasien temp = arrayPasien[i];  arrayPasien[i] = arrayPasien[j];  arrayPasien[j] = temp;  }  j += 1;  return rekurcionSwapDataTinggiBadan(j, high, pivot, i);  }  void quickSortBerdasarkanNamaLengkap(int low, int high) {  if (jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return;  }  if (low < high) {  int pivotIndex = partitionNamaLengkap(low, high);  quickSortBerdasarkanNamaLengkap(low, pivotIndex - 1);  quickSortBerdasarkanNamaLengkap(pivotIndex + 1, high);  }  }  private int partitionNamaLengkap(int low, int high) {  String pivot = arrayPasien[high].getNamaLengkap();  int i = low - 1;  i = rekurcionSwapDataNamaLengkap(low, high, pivot, i);  Pasien temp = arrayPasien[i + 1];  arrayPasien[i + 1] = arrayPasien[high];  arrayPasien[high] = temp;  return i + 1;  }  private int rekurcionSwapDataNamaLengkap(int j, int high, String pivot, int i) {  if (j >= high) {  return i;  }  String data = arrayPasien[j].getNamaLengkap();  if (data.compareToIgnoreCase(pivot) < 0) {  i++;  Pasien temp = arrayPasien[i];  arrayPasien[i] = arrayPasien[j];  arrayPasien[j] = temp;  }  j += 1;  return rekurcionSwapDataNamaLengkap(j, high, pivot, i);  }  void printSemuaDataPasien(int i) {  if (jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return;  }  if (i >= jumlahPasien) {  return;  }  printDataPasien(arrayPasien[i]);  printSemuaDataPasien(++i);  }  void printDataPasien(Pasien pasien) {  if (jumlahPasien == 0) {  System.out.println("Rumah Sakit Belum Memiliki Pasien");  return;  }  System.out.println("-------------------------------------------------");  System.out.println("Nama Lengkap : " + pasien.getNamaLengkap());  System.out.println("Tanggal Lahir : " + pasien.getTanggalLahir());  System.out.println("Riwayat Penyakit : " + pasien.getRiwayatPenyakit());  System.out.println("Golongan Darah : " + pasien.getGolonganDarah());  System.out.println("Dokter Penanggung Jawab : " + pasien.getDokterPenanggungJawab());  System.out.println("Berat Badan : " + pasien.getBeratBadan() + " Kg");  System.out.println("Tinggi Badan : " + pasien.getTinggiBadan() + " Cm");  System.out.println("-------------------------------------------------");  }  } |

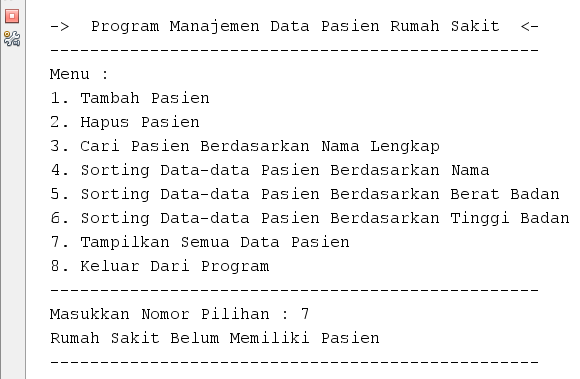
1. Pasien.java

|  |
| --- |
| package utp;  import java.util.Scanner;  public class Pasien {  private String namaLengkap;  private String tanggalLahir;  private String riwayatPenyakit;  private String golonganDarah;  private String dokterPenanggungJawab;  private double beratBadan;  private double tinggiBadan;  public String getNamaLengkap() {  return namaLengkap;  }  public void setNamaLengkap(String namaLengkap) {  this.namaLengkap = namaLengkap;  }  public String getTanggalLahir() {  return tanggalLahir;  }  public void setTanggalLahir(String tanggalLahir) {  this.tanggalLahir = tanggalLahir;  }  public String getRiwayatPenyakit() {  return riwayatPenyakit;  }  public void setRiwayatPenyakit(String riwayatPenyakit) {  this.riwayatPenyakit = riwayatPenyakit;  }  public String getGolonganDarah() {  return golonganDarah;  }  public void setGolonganDarah(String golonganDarah) {  this.golonganDarah = golonganDarah;  }  public String getDokterPenanggungJawab() {  return dokterPenanggungJawab;  }  public void setDokterPenanggungJawab(String dokterPenanggungJawab) {  this.dokterPenanggungJawab = dokterPenanggungJawab;  }  public double getBeratBadan() {  return beratBadan;  }  public void setBeratBadan(double beratBadan) {  this.beratBadan = beratBadan;  }  public double getTinggiBadan() {  return tinggiBadan;  }  public void setTinggiBadan(double tinggiBadan) {  this.tinggiBadan = tinggiBadan;  }  public void isiDataPasien() {  Scanner scan = new Scanner(System.in);  System.out.println("Masukkan Data");  System.out.print("-> Nama Lengkap : ");  setNamaLengkap(scan.nextLine());  System.out.print("-> Tanggal Lahir : ");  setTanggalLahir(scan.nextLine());  System.out.print("-> Riwayat Penyakit : ");  setRiwayatPenyakit(scan.nextLine());  System.out.print("-> Golongan Darah : ");  setGolonganDarah(scan.nextLine());  System.out.print("-> Dokter Penanggung Jawab: ");  setDokterPenanggungJawab(scan.nextLine());  System.out.print("-> Berat Badan : ");  setBeratBadan(scan.nextDouble());  System.out.print("-> Tinggi Badan : ");  setTinggiBadan(scan.nextDouble());  }      } |

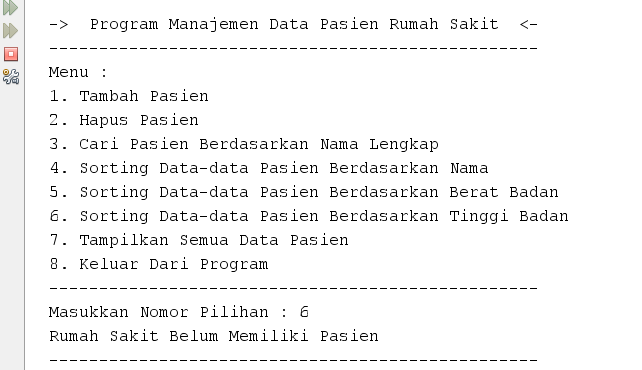
1. **Output**



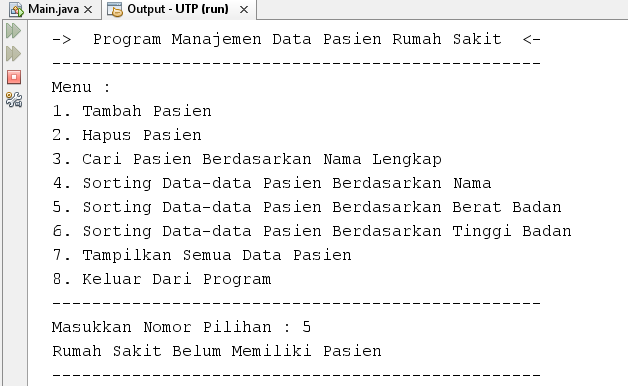
Gambar 3.1 hasil output



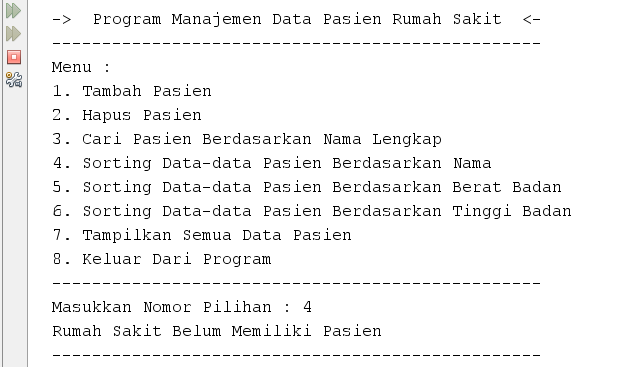
Gambar 3.2 hasil output jika piihan = 7 dan belum punya pasien (lanjutan gambar 3.1)



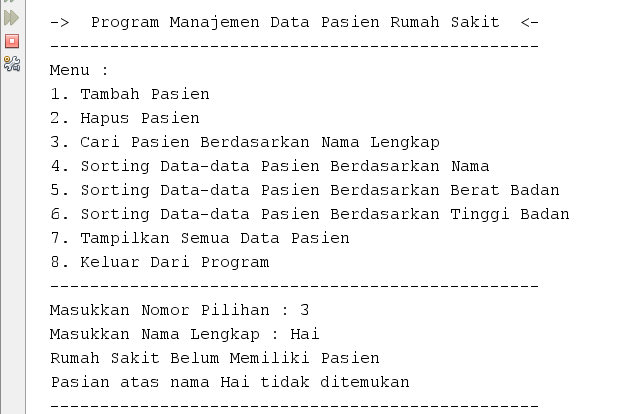
Gambar 3.3 hasil output jika piihan = 6 dan belum punya pasien (lanjutan gambar 3.2)



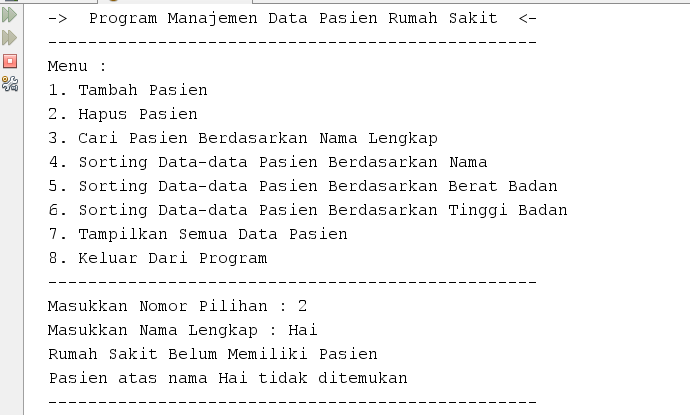
Gambar 3.4 hasil output jika piihan = 5 dan belum punya pasien (lanjutan gambar 3.3)



Gambar 3.5 hasil output jika piihan = 4 dan belum punya pasien (lanjutan gambar 3.4)



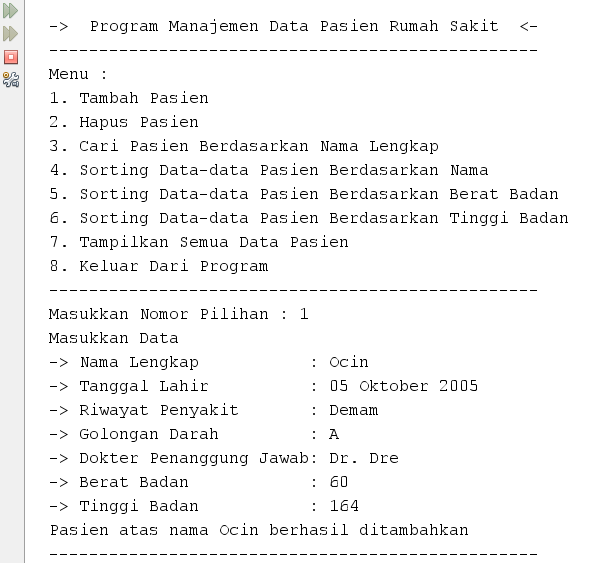
Gambar 3.6 hasil output jika piihan = 3, input nama = Hai dan belum punya pasien (lanjutan gambar 3.5)



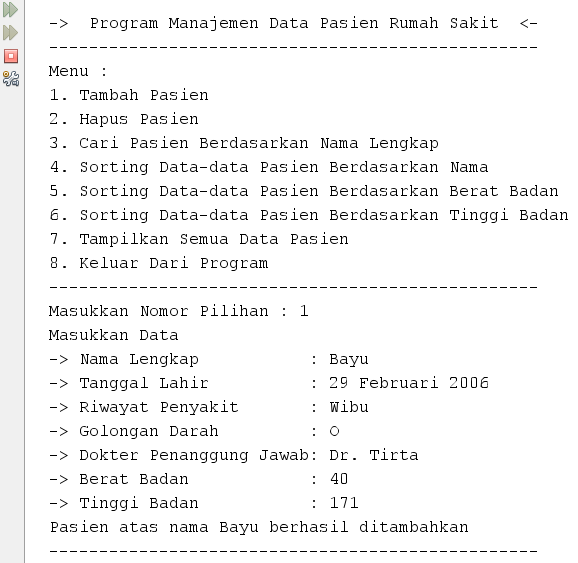
Gambar 3.7 hasil output jika piihan = 2, input nama = Hai dan belum punya pasien (lanjutan gambar 3.6)



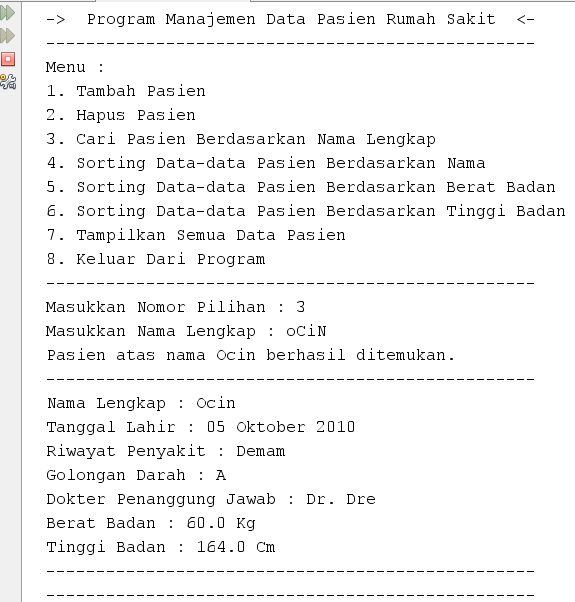
Gambar 3.8 hasil output jika piihan = 1, input nama lengkap = Azka, tanggal lahir = 21 Juni 2005, Riwayat Penyakit = Tipes, Gol. Darah = A , Dokter Penanggung Jawab = Dr. Dre, Berat Badan = 50, Tinggi Badan = 170 (lanjutan gambar 3.7)



Gambar 3.9 hasil output jika piihan = 1, input nama lengkap = Ocin, tanggal lahir = 05 Oktober 2010, Riwayat Penyakit = Demam, Gol. Darah = A , Dokter Penanggung Jawab = Dr. Dre, Berat Badan = 60, Tinggi Badan = 164 (lanjutan gambar 3.8)



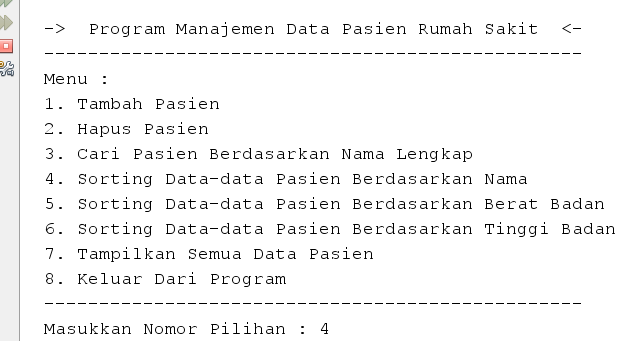
Gambar 3.10 hasil output jika piihan = 1, input nama lengkap = Bayu, tanggal lahir = 29 Februari 2006, Riwayat Penyakit = Wibu, Gol. Darah = O , Dokter Penanggung Jawab = Dr. Tirta, Berat Badan = 40, Tinggi Badan = 171 (lanjutan gambar 3.9)



Gambar 3.11 hasil output jika piihan = 3, input nama lengkap = oCiN (lanjutan gambar 3.10)



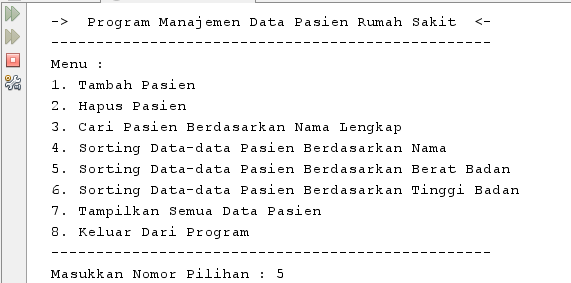
Gambar 3.12 hasil output jika piihan = 7 (lanjutan gambar 3.11)



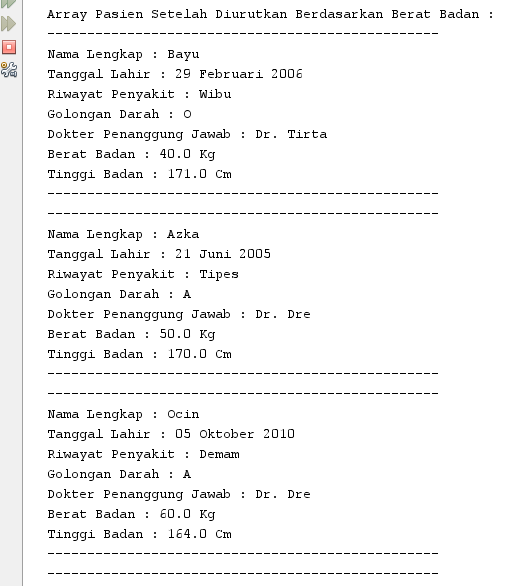
Gambar 3.13 hasil output jika piihan = 4 (lanjutan gambar 3.12)



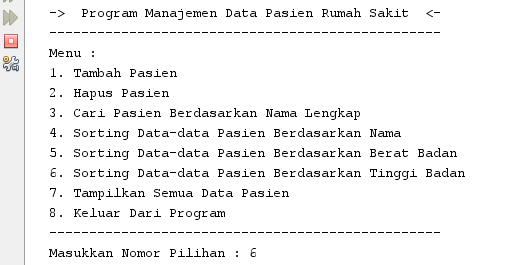
Gambar 3.14 hasil output (lanjutan gambar 3.13)



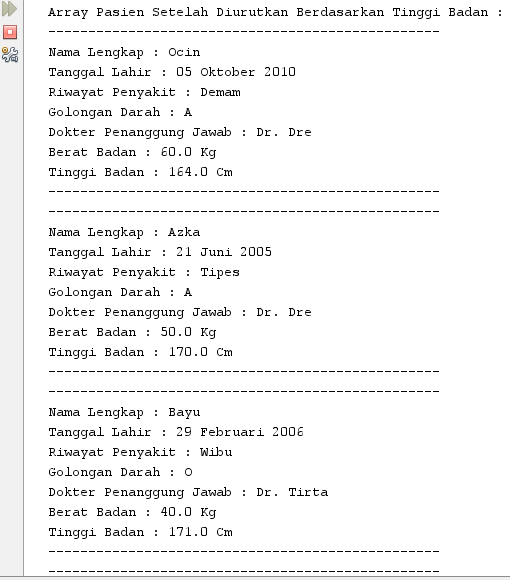
Gambar 3.15 hasil output jika piihan = 5 (lanjutan gambar 3.14)



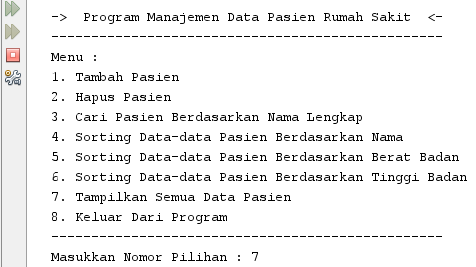
Gambar 3.16 hasil output (lanjutan gambar 3.15)



Gambar 3.17 hasil output jika piihan = 6 (lanjutan gambar 3.16)



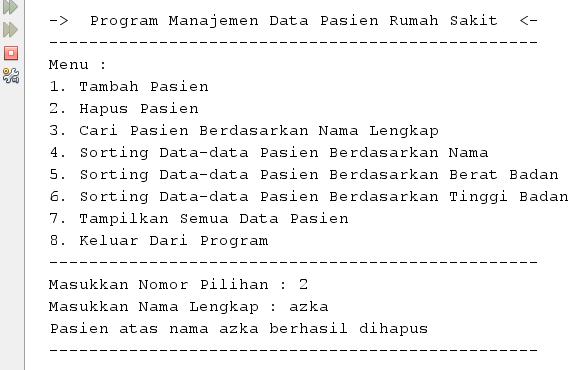
Gambar 3.18 hasil output (lanjutan gambar 3.17)

****

Gambar 3.19 hasil output jika pilihan = 7 (lanjutan gambar 3.18)

****

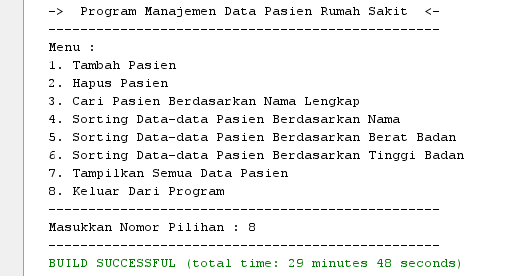
Gambar 3.20 hasil output (lanjutan gambar 3.19)



Gambar 3.21 hasil output jika pilihan = 7, dan nama lengkap = azka (lanjutan gambar 3.20)



Gambar 3.22 hasil output (lanjutan gambar 3.21)



Gambar 3.23 hasil output jika pilihan = 8 (lanjutan gambar 3.22)

**Sumber :**

Ali, I., Nawaz, H., Khan, I., Maitlo, A., Chhajro, M. A., & Rind, M. M. (2018). Performance comparison between merge and quick sort algorithms in data structure. *International Journal of Advanced Computer Science and Applications*, *9*(11).

Horstmann, C. S. (2017). *Big Java: Late Objects*. John Wiley & Sons.

Monoarfa, F., Prasetyo, Y., & Trianto, J. (2023). Aplikasi Manajemen Rumah Sakit MyHospital Berbasis Java. *Jurnal Maklumatika*, 31-41.

Saputra, B. D., & Stefanie, A. (2023). Automation testing api, android, dan website menggunakan serenity bdd pada software sistem manajemen rumah sakit. *Jurnal Ilmiah Wahana Pendidikan*, *9*(10), 114-126.

Setiawan, A. T., & Permadani, R. P. (2017). Sistem Informasi Manajemen Rumah Sakit Welasasih Ambarawa. *Jurnal TAM (Technology Acceptance Model)*, *7*, 52-60.