


J ManajemenPlaystation.java X

J ManajemenPlaystation.java >  ManajemenPlaystation

```
1  import java.util.ArrayList;
2  import java.util.Scanner;
3
4  class Pelanggan {
5      // Mengubah akses modifier menjadi private (encapsulation)
6      private String nama;
7      private String kontak;
8
9      public Pelanggan(String nama, String kontak) {
10         this.nama = nama;
11         this.kontak = kontak;
12     }
13
14     // Implementasi Getter
15     public String getNama() {
16         return nama;
17     }
18
19     public String getKontak() {
20         return kontak;
21     }
22
23     // Implementasi Setter
24     public void setNama(String nama) {
25         this.nama = nama;
26     }
27
28     public void setKontak(String kontak) {
29         this.kontak = kontak;
30     }
31
32     // Metode update menggunakan setter
33     public void updatePelanggan(String nama, String kontak) {
34         setNama(nama);
35         setKontak(kontak);
36     }
37
38     // Method untuk menampilkan info pelanggan - akan di-override oleh subclass
39     public String tampilInfo() {
40         return "Nama: " + getNama() + ", Kontak: " + getKontak();
41     }
42 }
43
44 // Menambahkan subclass PelangganMember untuk polimorfisme
45 class PelangganMember extends Pelanggan {
46     private String noMember;
47     private int poin;
48
49     // Constructor
50     public PelangganMember(String nama, String kontak, String noMember) {
51         super(nama, kontak);
52         this.noMember = noMember;
53         this.poin = 0;
54     }
55 }
```

```

54     }
55
56     // Constructor overloading
57     public PelangganMember(String nama, String kontak, String noMember, int poin) {
58         super(nama, kontak);
59         this.noMember = noMember;
60         this.poin = poin;
61     }
62
63     // Getter dan Setter
64     public String getNoMember() {
65         return noMember;
66     }
67
68     public void setNoMember(String noMember) {
69         this.noMember = noMember;
70     }
71
72     public int getPoin() {
73         return poin;
74     }
75
76     public void setPoin(int poin) {
77         this.poin = poin;
78     }
79
80     // Method overriding
81     @Override
82     public String tampilInfo() {
83         return super.tampilInfo() + ", No Member: " + noMember + ", Poin: " + poin;
84     }
85
86     // Method overloading
87     public void updatePelanggan(String nama, String kontak, String noMember) {
88         super.updatePelanggan(nama, kontak);
89         setNoMember(noMember);
90     }
91
92     // Method overloading
93     public void updatePelanggan(String nama, String kontak, String noMember, int poin) {
94         updatePelanggan(nama, kontak, noMember);
95         setPoin(poin);
96     }
97
98     // Method untuk menambah poin
99     public void tambahPoin(int jumlah) {
100         this.poin += jumlah;
101     }
102 }
103
104 class Konsol {
105     // Mengubah akses modifier menjadi private (encapsulation)
106     private String namaKonsol;
107     private boolean tersedia;
108
109     public Konsol(String namaKonsol, boolean tersedia) {

```

```

108
109     public Konsol(String namaKonsol, boolean tersedia) {
110         this.namaKonsol = namaKonsol;
111         this.tersedia = tersedia;
112     }
113
114     // Implementasi Getter
115     public String getNamaKonsol() {
116         return namaKonsol;
117     }
118
119     public boolean isTersedia() {
120         return tersedia;
121     }
122
123     // Implementasi Setter
124     public void setNamaKonsol(String namaKonsol) {
125         this.namaKonsol = namaKonsol;
126     }
127
128     public void setTersedia(boolean tersedia) {
129         this.tersedia = tersedia;
130     }
131
132     // Metode update menggunakan setter
133     public void updateKonsol(boolean tersedia) {
134         setTersedia(tersedia);
135     }
136
137     // Method yang akan di-override
138     public String infoKonsol() {
139         return "Nama: " + getNamaKonsol() + ", Tersedia: " + (isTersedia() ? "Ya" : "Tidak");
140     }
141 }
142
143 // Subclass untuk KonsolPS untuk implementasi polimorfisme
144 class KonsolPS5 extends Konsol {
145     private boolean digitalEdition;
146
147     public KonsolPS5(String namaKonsol, boolean tersedia, boolean digitalEdition) {
148         super(namaKonsol, tersedia);
149         this.digitalEdition = digitalEdition;
150     }
151
152     public boolean isDigitalEdition() {
153         return digitalEdition;
154     }
155
156     public void setDigitalEdition(boolean digitalEdition) {
157         this.digitalEdition = digitalEdition;
158     }
159
160     // Metode update

```

```

159
160     // Metode overriding
161     @Override
162     public String infoKonsol() {
163         return super.infoKonsol() + ", Digital Edition: " + (isDigitalEdition() ? "Ya" : "Tidak");
164     }
165
166     // Metode overloading
167     public void updateKonsol(boolean tersedia, boolean digitalEdition) {
168         updateKonsol(tersedia);
169         setDigitalEdition(digitalEdition);
170     }
171 }
172
173 class Karyawan {
174     // Mengubah akses modifier (encapsulation)
175     private String nama;
176     // Menggunakan protected untuk ID - bisa diakses oleh subclass
177     protected String id;
178
179     public Karyawan(String nama, String id) {
180         this.nama = nama;
181         this.id = id;
182     }
183
184     // Implementasi Getter
185     public String getNama() {
186         return nama;
187     }
188
189     public String getId() {
190         return id;
191     }
192
193     // Implementasi Setter
194     public void setNama(String nama) {
195         this.nama = nama;
196     }
197
198     public void setId(String id) {
199         this.id = id;
200     }
201
202     // Metode update menggunakan setter
203     public void updateKaryawan(String nama) {
204         setNama(nama);
205     }
206
207     // Method yang akan di-override
208     public double hitungGaji() {
209         return 3000000.0; // Gaji default karyawan
210     }

```

```

210     }
211
212     // Method untuk menampilkan info karyawan
213     public String infoKaryawan() {
214         return "Nama: " + getName() + ", ID: " + getId() + ", Gaji: Rp " + hitungGaji();
215     }
216 }
217
218 // Subclass untuk Manager
219 class Manager extends Karyawan {
220     private double bonus;
221
222     public Manager(String nama, String id, double bonus) {
223         super(nama, id);
224         this.bonus = bonus;
225     }
226
227     public double getBonus() {
228         return bonus;
229     }
230
231     public void setBonus(double bonus) {
232         this.bonus = bonus;
233     }
234
235     // Method overriding
236     @Override
237     public double hitungGaji() {
238         return super.hitungGaji() + bonus;
239     }
240
241     // Method overloading
242     public void updateKaryawan(String nama, double bonus) {
243         super.updateKaryawan(nama);
244         setBonus(bonus);
245     }
246 }
247
248 public class ManajemenPlaystation {
249     // Scanner sebagai private static field
250     private static Scanner scanner = new Scanner(System.in);
251     // Membuat daftar-daftar sebagai protected - bisa diakses oleh subclass
252     protected static ArrayList<Pelanggan> daftarPelanggan = new ArrayList<>();
253     protected static ArrayList<Konsol> daftarKonsol = new ArrayList<>();
254     protected static ArrayList<Karyawan> daftarKaryawan = new ArrayList<>();
255

```

Run | Debug

```
256 public static void main(String[] args) {
257     boolean running = true;
258     while (running) {
259         System.out.println(x:"\n=== Manajemen Playstation ===");
260         System.out.println(x:"1. Tambah Pelanggan");
261         System.out.println(x:"2. Tambah Pelanggan Member");
262         System.out.println(x:"3. Tambah Konsol");
263         System.out.println(x:"4. Tambah Konsol PS5");
264         System.out.println(x:"5. Lihat Data Pelanggan");
265         System.out.println(x:"6. Lihat Data Konsol");
266         System.out.println(x:"7. Update Data Pelanggan");
267         System.out.println(x:"8. Update Data Konsol");
268         System.out.println(x:"9. Hapus Data Pelanggan");
269         System.out.println(x:"10. Hapus Data Konsol");
270         System.out.println(x:"11. Tambah Karyawan");
271         System.out.println(x:"12. Tambah Manager");
272         System.out.println(x:"13. Update Data Karyawan");
273         System.out.println(x:"14. Lihat Data Karyawan");
274         System.out.println(x:"15. Hapus Data Karyawan");
275         System.out.println(x:"16. Keluar");
276         System.out.print(s:"Pilih menu: ");
277         int pilihan = scanner.nextInt();
278         scanner.nextLine();
279
280         switch (pilihan) {
281             case 1:
282                 tambahPelanggan();
283                 break;
284             case 2:
285                 tambahPelangganMember();
286                 break;
287             case 3:
288                 tambahKonsol();
289                 break;
290             case 4:
291                 tambahKonsolPS5();
292                 break;
293             case 5:
294                 lihatPelanggan();
295                 break;
296             case 6:
297                 lihatKonsol();
298                 break;
299             case 7:
300                 updatePelanggan();
301                 break;
302             case 8:
303                 updateKonsol();
304                 break;
305             case 9:
306                 hapusPelanggan();
307                 break;
308             case 10:
309                 hapusKonsol();
310                 break;
```

```

308         case 10:
309             hapusKonsol();
310             break;
311         case 11:
312             tambahKaryawan();
313             break;
314         case 12:
315             tambahManager();
316             break;
317         case 13:
318             updateKaryawan();
319             break;
320         case 14:
321             lihatKaryawan();
322             break;
323         case 15:
324             hapusKaryawan();
325             break;
326         case 16:
327             running = false;
328             break;
329         default:
330             System.out.println(x:"Pilihan tidak valid, coba lagi.");
331     }
332 }
333
334
335 // Metode dengan akses modifier public
336 public static void tambahPelanggan() {
337     System.out.print(s:"Masukkan nama pelanggan: ");
338     String nama = scanner.nextLine();
339     System.out.print(s:"Masukkan kontak pelanggan: ");
340     String kontak = scanner.nextLine();
341     daftarPelanggan.add(new Pelanggan(nama, kontak));
342     System.out.println(x:"Pelanggan berhasil ditambahkan!");
343 }
344
345 // Metode untuk tambah pelanggan member (implementation of polymorphism)
346 public static void tambahPelangganMember() {
347     System.out.print(s:"Masukkan nama pelanggan member: ");
348     String nama = scanner.nextLine();
349     System.out.print(s:"Masukkan kontak pelanggan member: ");
350     String kontak = scanner.nextLine();
351     System.out.print(s:"Masukkan nomor member: ");
352     String noMember = scanner.nextLine();
353     System.out.print(s:"Masukkan poin awal (0 jika baru): ");
354     int poin = scanner.nextInt();
355     scanner.nextLine();
356
357     daftarPelanggan.add(new PelangganMember(nama, kontak, noMember, poin));
358     System.out.println(x:"Pelanggan Member berhasil ditambahkan!");
359 }
360

```

```

360
361 public static void lihatPelanggan() {
362     if (daftarPelanggan.isEmpty()) {
363         System.out.println(x:"Belum ada pelanggan.");
364     } else {
365         System.out.println(x:"\nDaftar Pelanggan:");
366         for (int i = 0; i < daftarPelanggan.size(); i++) {
367             Pelanggan p = daftarPelanggan.get(i);
368             // Menggunakan method polymorphic
369             System.out.println((i + 1) + ". " + p.tampilInfo());
370         }
371     }
372 }
373
374 public static void updatePelanggan() {
375     lihatPelanggan();
376     if (daftarPelanggan.isEmpty()) return;
377     System.out.print(s:"Pilih nomor pelanggan yang ingin diupdate: ");
378     int index = scanner.nextInt() - 1;
379     scanner.nextLine();
380     if (index >= 0 && index < daftarPelanggan.size()) {
381         Pelanggan p = daftarPelanggan.get(index);
382
383         System.out.print(s:"Masukkan nama baru: ");
384         String nama = scanner.nextLine();
385         System.out.print(s:"Masukkan kontak baru: ");
386         String kontak = scanner.nextLine();
387
388         // Polymorphism - different behavior based on object type
389         if (p instanceof PelangganMember) {
390             System.out.print(s:"Masukkan nomor member baru: ");
391             String noMember = scanner.nextLine();
392             System.out.print(s:"Masukkan poin baru: ");
393             int poin = scanner.nextInt();
394             scanner.nextLine();
395             ((PelangganMember) p).updatePelanggan(nama, kontak, noMember, poin);
396         } else {
397             p.updatePelanggan(nama, kontak);
398         }
399
400         System.out.println(x:"Data pelanggan berhasil diperbarui!");
401     } else {
402         System.out.println(x:"Nomor tidak valid.");
403     }
404 }
405
406 public static void tambahKonsol() {
407     System.out.print(s:"Masukkan nama konsol: ");
408     String namaKonsol = scanner.nextLine();
409     daftarKonsol.add(new Konsol(namaKonsol, tersedia:true));
410     System.out.println(x:"Konsol berhasil ditambahkan!");
411 }

```



```

413 // Metode untuk menambah konsol PS5
414 public static void tambahKonsolPS5() {
415     System.out.print(s:"Masukkan nama konsol PS5: ");
416     String namaKonsol = scanner.nextLine();
417     System.out.print(s:"Tersedia? (true/false): ");
418     boolean tersedia = scanner.nextBoolean();
419     System.out.print(s:"Digital Edition? (true/false): ");
420     boolean digitalEdition = scanner.nextBoolean();
421     scanner.nextLine();
422
423     daftarKonsol.add(new KonsolPS5(namaKonsol, tersedia, digitalEdition));
424     System.out.println(x:"Konsol PS5 berhasil ditambahkan!");
425 }
426
427 public static void lihatKonsol() {
428     if (daftarKonsol.isEmpty()) {
429         System.out.println(x:"Belum ada konsol.");
430     } else {
431         System.out.println(x:"\nDaftar Konsol:");
432         for (int i = 0; i < daftarKonsol.size(); i++) {
433             Konsol k = daftarKonsol.get(i);
434             // Menggunakan polymorphic method
435             System.out.println((i + 1) + ". " + k.infoKonsol());
436         }
437     }
438 }
439
440 public static void updateKonsol() {
441     lihatKonsol();
442     if (daftarKonsol.isEmpty()) return;
443     System.out.print(s:"Pilih nomor konsol yang ingin diupdate: ");
444     int index = scanner.nextInt() - 1;
445     scanner.nextLine();
446     if (index >= 0 && index < daftarKonsol.size()) {
447         Konsol k = daftarKonsol.get(index);
448         System.out.print(s:"Apakah konsol tersedia? (true/false): ");
449         boolean tersedia = scanner.nextBoolean();
450
451         // Polymorphism - different behavior based on object type
452         if (k instanceof KonsolPS5) {
453             System.out.print(s:"Digital Edition? (true/false): ");
454             boolean digitalEdition = scanner.nextBoolean();
455             ((KonsolPS5) k).updateKonsol(tersedia, digitalEdition);
456         } else {
457             k.updateKonsol(tersedia);
458         }
459
460         scanner.nextLine();
461         System.out.println(x:"Konsol berhasil diperbarui!");
462     } else {
463         System.out.println(x:"Nomor tidak valid.");
464     }
465 }

```

```

465     }
466
467     public static void hapusPelanggan() {
468         lihatPelanggan();
469         if (daftarPelanggan.isEmpty()) return;
470         System.out.print(s:"Pilih nomor pelanggan yang ingin dihapus: ");
471         int index = scanner.nextInt() - 1;
472         scanner.nextLine();
473         if (index >= 0 && index < daftarPelanggan.size()) {
474             daftarPelanggan.remove(index);
475             System.out.println(x:"Pelanggan berhasil dihapus!");
476         } else {
477             System.out.println(x:"Nomor tidak valid.");
478         }
479     }
480
481     public static void hapusKonsol() {
482         lihatKonsol();
483         if (daftarKonsol.isEmpty()) return;
484         System.out.print(s:"Pilih nomor konsol yang ingin dihapus: ");
485         int index = scanner.nextInt() - 1;
486         scanner.nextLine();
487         if (index >= 0 && index < daftarKonsol.size()) {
488             daftarKonsol.remove(index);
489             System.out.println(x:"Konsol berhasil dihapus!");
490         } else {
491             System.out.println(x:"Nomor tidak valid.");
492         }
493     }
494
495     public static void tambahKaryawan() {
496         System.out.print(s:"Masukkan nama karyawan: ");
497         String nama = scanner.nextLine();
498         System.out.print(s:"Masukkan ID karyawan: ");
499         String id = scanner.nextLine();
500         daftarKaryawan.add(new Karyawan(nama, id));
501         System.out.println(x:"Karyawan berhasil ditambahkan!");
502     }
503
504     // Metode untuk menambah manager
505     public static void tambahManager() {
506         System.out.print(s:"Masukkan nama manager: ");
507         String nama = scanner.nextLine();
508         System.out.print(s:"Masukkan ID manager: ");
509         String id = scanner.nextLine();
510         System.out.print(s:"Masukkan bonus manager: ");
511         double bonus = scanner.nextDouble();
512         scanner.nextLine();
513
514         daftarKaryawan.add(new Manager(nama, id, bonus));
515         System.out.println(x:"Manager berhasil ditambahkan!");
516     }
517

```

```

516     }
517
518     public static void lihatKaryawan() {
519         if (daftarKaryawan.isEmpty()) {
520             System.out.println(x: "Belum ada karyawan.");
521         } else {
522             System.out.println(x: "\nDaftar Karyawan:");
523             for (int i = 0; i < daftarKaryawan.size(); i++) {
524                 Karyawan k = daftarKaryawan.get(i);
525                 // Menggunakan method polymorphic
526                 System.out.println((i + 1) + ". " + k.infoKaryawan());
527             }
528         }
529     }
530
531     public static void updateKaryawan() {
532         lihatKaryawan();
533         if (daftarKaryawan.isEmpty()) return;
534         System.out.print(s: "Pilih nomor karyawan yang ingin diupdate: ");
535         int index = scanner.nextInt() - 1;
536         scanner.nextLine();
537         if (index >= 0 && index < daftarKaryawan.size()) {
538             Karyawan k = daftarKaryawan.get(index);
539             System.out.print(s: "Masukkan nama baru: ");
540             String nama = scanner.nextLine();
541
542             // Polymorphism - different behavior based on object type
543             if (k instanceof Manager) {
544                 System.out.print(s: "Masukkan bonus baru: ");
545                 double bonus = scanner.nextDouble();
546                 scanner.nextLine();
547                 ((Manager) k).updateKaryawan(nama, bonus);
548             } else {
549                 k.updateKaryawan(nama);
550             }
551
552             System.out.println(x: "Data karyawan berhasil diperbarui!");
553         } else {
554             System.out.println(x: "Nomor tidak valid.");
555         }
556     }
557
558     public static void hapusKaryawan() {
559         lihatKaryawan();
560         if (daftarKaryawan.isEmpty()) return;
561         System.out.print(s: "Pilih nomor karyawan yang ingin dihapus: ");
562         int index = scanner.nextInt() - 1;
563         scanner.nextLine();
564         if (index >= 0 && index < daftarKaryawan.size()) {
565             daftarKaryawan.remove(index);
566             System.out.println(x: "Karyawan berhasil dihapus!");
567         } else {
568             System.out.println(x: "Nomor tidak valid.");
569         }
570     }
571 }

```

