switch Deyimi

switch deyiminin genel biçimi şöyledir:

```
switch (<tam sayı türünden ifade>) {
	case <sabit ifadesi(s.i)>:
		//...
		[break;]
	case <s.i>:
		//...
		[break;]
	case <s.i>:
		//...
		[break;]
	//...
	[
	default:
		//...
		break;
	]
}
```

switch anahtar sözcüğünden sonra parantezler içerisinde bir ifade bulunmak zorundadır. switch deyimi case bölümlerinden oluşur. case anahtar sözcüğünden sonra bir sabit ifadesi bulunmak zorundadır. switch deyimi default bölüme sahip olabilir.

Anahtar Notlar: Yalnızca operatör ve sabitlerden oluşan ifadelere sabit ifadeleri denir. Tek başına sabitler de sabit ifadesidir. Örneğin:

```
8
10 - 8 * 11
11 - 8
```

birer sabit ifadesidir. Java 'da bazı durumlarda sabit ifadesi kullanmak zorunludur. Sabit ifadelerinin net sayısal değerleri derleme aşamasında belirlenip (constant folding) byte koda yazılmaktadır.

switch deyimi şöyle çalışır:

Önce switch deyimi içerisindeki ifadenin sayısal değeri hesaplanır.Sonra bu değere tam eşit olan case bölümü var mı diye tek tek bakılır. Eğer böyle bir case bölümü varsa akış o bölüme aktarılır. break deyimi switch deyimini sonlandırmak için de kullanılabilmektedir. Eğer switch parantezi içerisindeki ifade ile aynı değerde bir case bölümü yoksa fakat default bölüm varsa akış default bölümüne aktarılır. default bölümü yoksa akış switch deyiminin sonundan devam eder.

Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
```

```
		java.util.Scanner kb = new java.util.Scanner(System.in);

		System.out.println("Bir sayı giriniz");
		int val = Integer.parseInt(kb.nextLine());

		switch (val) {
		case 1:
			System.out.println("Bir");
			break;
		case 2:
			System.out.println("İki");
			break;
		default:
			System.out.println("Geçersiz Sayı");
		}

		kb.close();
	}
}
```

switch deyimi içerisinde aynı değerde birden fazla case bölümü bulunamaz. Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		java.util.Scanner kb = new java.util.Scanner(System.in);

		System.out.println("Bir sayı giriniz");
		int val = Integer.parseInt(kb.nextLine());

		switch (val) {
		case 1:
			System.out.println("Bir");
			break;
		case 2: //error:
			System.out.println("İki");
			break;
		case 4 / 2: //error: Dikkat case 2: ile aynı
			break;
		default:
			System.out.println("Geçersiz Sayı");
		}

		kb.close();
	}
}
```

Sabit ifadelerinin değerleri derleme zamanında hesaplanabildiğinden case 2: ile case 4 / 2: bölümleri aynı değerde olacaklardır.

case bölümlerinin sıralı olarak dizilmesi ya da default bölümün sonda olması zorunlu değildir. Ancak hem performans hem de okunabilirlik açısından mümkünse gelme olasılığı daha yüksek olan değerler üste, default bölümü tüm case bölümlerinin altına yazılmalıdır.

switch parantezi içerisindeki ifade ve case ifadeleri tamsayı türlerine ilişkin olmak zorundadır. Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
```

```
	{
		java.util.Scanner kb = new java.util.Scanner(System.in);

		System.out.println("Bir sayı giriniz");
		double val = Double.parseDouble(kb.nextLine());

		switch (val) { // error
		case 1:
			System.out.println("Bir");
			break;
		case 2:
			System.out.println("İki");
			break;
		default:
			System.out.println("Geçersiz Sayı");
		}

		kb.close();
	}
}
```

Burada val double türden olduğundan geçersizdir.

Java'da switch deyiminin aşağı düşme (fall through) özelliği bulunmaktadır. Buna göre akış herhangi bir case bölümüne geldiğinde o akışı yönlendirecek herhangi bir deyim görene kadar tüm deyimler çalıştırılacakır. Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		java.util.Scanner kb = new java.util.Scanner(System.in);

		System.out.println("Bir sayı giriniz");
		int val = Integer.parseInt(kb.nextLine());


		switch (val) {
		case 1:
			System.out.println("Bir");
			break;
		case 2:
			System.out.println("İki");
		case 3:
			System.out.println("Üç");
		case 4:
			System.out.println("Dört");
			break;
		default:
			System.out.println("Geçersiz Sayı");
		}

		kb.close();
	}
}
```

Burada örneğin kullanıcı klavyeden 2 (iki) değerini girerse aşağı düşme özellğinden dolayı case2, case 3 ve case 4 bölümlerinin hepsine ilişkin deyimler çalıştırılacaktır. Aşağı düşmeyi engellemek için break deyimi kullanılabilir. Bununla birlikte akışı aşağı düşürmeyecek herhangi bir deyim de (return, continue, etiktetli break vb.) kullanılabilir. Bazen programcı aşağı düşme özellğinden faydalanabilir. Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		java.util.Scanner kb = new java.util.Scanner(System.in);

		System.out.println("Seçenek?");
		char ch = kb.nextLine().charAt(0);

		switch (ch) {
		case 'e':
		case 'E':
			System.out.println("Ekle");
			break;
		case 's':
		case 'S':
			System.out.println("Sil");
			break;
		default:
			System.out.println("Geçersiz Menü");
		}

		kb.close();
	}
}
```

Burada kullanıcı örneğin E veya e girdiğinde aşağı düşme özellği sayesinde aynı işlem yapılacaktır. Bu işlem en kısa bu şekilde yapılabilir.

Örnek Program:
Basit bir menü uygulaması:

```
package csd;

class App {
	public static void displayMenu()
	{
		System.out.println("1.Ekle");
		System.out.println("2.Güncelle");
		System.out.println("3.Sil");
		System.out.println("4.Listele");
		System.out.println("5.Çıkış");
		System.out.print("Seçenek:");
	}

	public static void main(String[] args)
	{
		java.util.Scanner kb = new java.util.Scanner(System.in);

		MENU:
		for (;;) {
			displayMenu();
			int option = Integer.parseInt(kb.nextLine());

			if (option < 1 || option > 5) {
				System.out.println("Geçersiz Seçenek");
				continue;
			}

			switch (option) {
			case 1:
				System.out.println("Ekle");
				break;
			case 2:
```

```
                        System.out.println("Güncelle");
                        break;
                case 3:
                        System.out.println("Sil");
                        break;
                case 4:
                        System.out.println("Listele");
                        break;
                case 5:
                        break MENU;
                }
        }

        System.out.println("Teşekkürler");

        kb.close();
    }
}
```

Dikkat edilirse sonsuz döngü içerisindeki switch deyiminden döngü dışına tek hamlede çıkmak için etiketli break kullanılmıştır. Aynı uygulama default bölümü kullanılarak da yapılabilir. Örneğin:

```
package csd;

class App {
    public static void displayMenu()
    {
        System.out.println("1.Ekle");
        System.out.println("2.Güncelle");
        System.out.println("3.Sil");
        System.out.println("4.Listele");
        System.out.println("5.Çıkış");
        System.out.print("Seçenek:");
    }

    public static void main(String[] args)
    {
        java.util.Scanner kb = new java.util.Scanner(System.in);

        MENU:
        for (;;) {
            displayMenu();
            int option = Integer.parseInt(kb.nextLine());

            switch (option) {
            case 1:
                System.out.println("Ekle");
                break;
            case 2:
                System.out.println("Güncelle");
                break;
            case 3:
                System.out.println("Sil");
                break;
            case 4:
                System.out.println("Listele");
                break;
            case 5:
                break MENU;
            default:
                System.out.println("Geçersiz Seçenek");
            }
        }

        System.out.println("Teşekkürler");
```

```
		kb.close();
	}
}
```

Programın okunabilirliği/anlaşılabilirliği açısından case bölümleri içerisinde çok fazla kod yazılması tavsiye edilmez. Bu durumda metot yazılması ve case bölümünde çağrılması iyi bir tekniktir. Örneğin yukarıdaki basit menü uygulaması metotlar kullanılarak şu şekilde yapılabilir:

```
package csd;

class App {
	public static void displayMenu()
	{
		System.out.println("1.Ekle");
		System.out.println("2.Güncelle");
		System.out.println("3.Sil");
		System.out.println("4.Listele");
		System.out.println("5.Çıkış");
		System.out.print("Seçenek:");
	}

	public static void addProc()
	{
		System.out.println("Ekle");
		//...
	}

	public static void updateProc()
	{
		System.out.println("Güncelle");
		//...
	}

	public static void deleteProc()
	{
		System.out.println("Sil");
		//...
	}

	public static void listProc()
	{
		System.out.println("Listele");
		//...
	}

	public static void main(String[] args)
	{
		java.util.Scanner kb = new java.util.Scanner(System.in);

		MENU:
		for (;;) {
			displayMenu();
			int option = Integer.parseInt(kb.nextLine());

			if (option < 1 || option > 5) {
				System.out.println("Geçersiz Seçenek");
				continue;
			}
			switch (option) {
			case 1:
				addProc();
				break;
			case 2:
				updateProc();
```

```
                        break;
                case 3:
                        deleteProc();
                        break;
                case 4:
                        listProc();
                        break;
                case 5:
                        break MENU;
                }
        }

        System.out.println("Teşekkürler");

        kb.close();
    }
}
```