break Deyimi

break deyiminin genel biçimi şöyledir:

break [etiket ismi];

break deyimi döngülerin içerisinde ya da switch deyimi içerisinde kullanılır. break deyimin tek başına kullanımında programın akışı break deyimine geldiğinde döngü sonlandırılır. Akış döngüden sonraki deyimle 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("Sayıları girmeye başlayınız");

		int val;
		int sum = 0;

		for (;;) {
			val = Integer.parseInt(kb.nextLine());
			if (val == 0)
				break;

			sum += val;
		}

		System.out.printf("Toplam:%d\n", sum);

		kb.close();
	}
}
```

İç içe döngülerde içteki döngüdeki break deyimi yalnızca kendi döngüsünü sonlandırır. İçteki döngü içerisinde tek hamlede dıştaki döngünün de sonlandırılması isteniyorsa bayrak (flag) değişken kullanılabilir.

Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		boolean exitFlag = false;

		for (int i = -12; i < 20; ++i) {
			for (int k = 5; k < 30; ++k) {
				if (i % k == 0) {
					exitFlag = true;
					break;
				}
				System.out.printf("(%d, %d)\n", i, k);
			}
			if (exitFlag)
				break;
		}
	}
}
```

Ya da örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		boolean exitFlag = false;

		for (int i = -12; i < 20; ++i) {
			for (int j = -5; j < 30; ++j) {
				for (int k = 5; k < 30; ++k) {
					if ((i + j + k) % 5 == 0) {
						exitFlag = true;
						break;
					}
					System.out.printf("(%d, %d, %d)\n", i, j, k);
				}
				if (exitFlag)
					break;
				//...
			}
			if (exitFlag)
				break;
			//...
		}
	}
}
```

break deyimi bir etiket ile de kullanılabilir. Buna etiketli break (labelled break) denilmektedir. Etiket değişken isimlendirme kurallarına uygun olarak bildirilmelidir. Etiket isimleri okunabilirlik/algılanabilirlik açısından tüm harfleri büyük olarak ve kelimeler arasında alttire karakteri konularak bildirilmelidir. Etiketli break deyimi iç içe döngülerden ya da döngü içerisindeki switch deyiminden tek hamlede çıkmak için kullanılabilir. Etiket sonlandırılacak döngü deyiminden hemen önce bildirilmelidir. Örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		LOOP_EXIT:
		for (int i = -12; i < 20; ++i) {
			for (int k = 5; k < 30; ++k) {
				if (i % k == 0)
					break LOOP_EXIT;
				System.out.printf("(%d, %d)\n", i, k);
			}
		}

		System.out.println("Program buradan devam ediyor");
	}
}
```

Ya da örneğin:

```
package csd;

class App {
	public static void main(String[] args)
	{
		LOOP_EXIT:
		for (int i = -12; i < 20; ++i) {
			for (int j = -5; j < 30; ++j) {
```

```
                        for (int k = 5; k < 30; ++k) {
                                if ((i + j + k) % 5 == 0) {
                                        break LOOP_EXIT;
                                }
                                System.out.printf("(%d, %d, %d)\n", i, j, k);
                        }
                }
        }
    }
}
```

Aşağıdaki programda en içteki döngü içerisinde klavyeden girilen sayı asalsa en dıştaki döngüden, pozitifse en içteki döngüden, negatifse ortadaki döngüden etiketli break deyimi kullanılarak çıkılmaktadır. Sıfır girilirse akış devam etmektedir:

```
package csd;

class App {
        public static boolean isPrime(int val)
        {
                if (val == 2)
                        return true;

                if (val <= 1 || val % 2 == 0)
                        return false;

                if (val == 3 || val == 5 || val == 7)
                        return true;

                for (int i = 3; i * i <= val; i += 2)
                        if (val % i == 0)
                                return false;

                return true;
        }

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

                THIRD_LOOP:
                for (int i = 0; i < 10; ++i) {
                        SECOND_LOOP:
                        for (int j = 0; j < 20; ++j) {
                                for (int k = 0; k < 30; ++k) {
                                        System.out.print("Sayı?");
                                        int number = Integer.parseInt(kb.nextLine());

                                        if (isPrime(number))
                                                break THIRD_LOOP;

                                        if (number > 0)
                                                break;

                                        if (number < 0)
                                                break SECOND_LOOP;

                                        System.out.println("İçteki döngü");
                                }
                                System.out.println("Ortadaki döngü");
                        }
                        System.out.println("Dıştaki döngü");
                }
```

```
            System.out.println("Program sonu");

            kb.close();
        }
}
```

Anahtar Notlar: Java' da goto deyimi yoktur. Etiketli break deyimi goto deyimine benzetilebilir. goto deyiminin (dolayısıyla etiketli break deyiminin) yapısal/nesne yönelimli programlamada olur olmaz yerde kullanılmaması gerekir. Etiketli break deyimi Java' da yalnızca içiçe döngülerden ve döngü deyimi içerisindeki switch deyiminden tek hamlede çıkmak için kullanılmalıdır.