**For Loops**

Ketika menulis program komputer, akan ada situasi di mana kita perlu melakukan hal sama berkali-kali. Misalnya kita ingin menampilkan semua nama pengguna yang terdaftar di aplikasi kita, atau sesederhana menampilkan angka 1 sampai 10. Tentunya tidak praktis jika kita menulis kode seperti berikut:

1. print(1);
2. print(2);
3. print(3);
4. print(4);
5. print(5);
6. print(6);
7. print(7);
8. print(8);
9. print(9);
10. print(10);

Bagaimana jika kita perlu menampilkan angka 1 sampai 100?

Dart memiliki banyak opsi untuk melakukan perulangan kode, salah satunya adalah for. For cocok digunakan pada kondisi perulangan di mana kita membutuhkan variabel indeks dan tahu berapa kali perulangan yang kita butuhkan. Sebagai contoh jika kita ingin menampilkan angka 1 sampai 100, kita bisa menuliskan seperti berikut:

1. for (int i = 1; i <= 100; i++) {
2. print(i);
3. }

Lebih ringkas bukan? Terdapat tiga bagian utama dalam sintaks for di atas:

* Pertama, variabel *index* yang seringkali diberi nama i yang berarti *index*. Pada variabel ini kita menginisialisasi nilai awal dari perulangan yang kita lakukan.
* Kedua, operasi perbandingan. Pada bagian ini komputer akan melakukan pengecekan kondisi apakah perulangan masih perlu dilakukan. Jika bernilai true maka kode di dalam blok for akan dijalankan.
* Ketiga, *increment/decrement*. Di sini kita melakukan penambahan atau pengurangan variabel *index*. Jadi pada contoh di atas variabel indeks akan ditambah dengan 1 di setiap akhir perulangan.

Jika dituliskan dalam bentuk *pseudocode*, maka kode di atas bisa dimaknai dengan “Jika i kurang dari sama dengan 100, maka jalankan kode berikut.”

**Challenge**

Kini saatnya menguji pemahaman Anda tentang materi *for loops*. Bisakah Anda membuat program Dart yang menampilkan output seperti berikut?

1. \*
2. \*\*
3. \*\*\*
4. \*\*\*\*
5. \*\*\*\*\*
6. \*\*\*\*\*\*
7. \*\*\*\*\*\*\*
8. \*\*\*\*\*\*\*\*
9. \*\*\*\*\*\*\*\*\*
10. \*\*\*\*\*\*\*\*\*\*