Transformation Towards the Ultimate

Kampus Merdeka INDONESIA JAYA

TIF1101 – Dasar-Dasar Pemrograman

HO 08 - Struktur Perulangan

Opim Salim Sitompul

Department of Information Technology
Universitas Sumatera Utara

Outline

Pendahuluan

- Struktur perulangan adalah salah satu struktur bahasa pemrograman yang memungkinkan pemrogram untuk melaksanakan satu atau lebih pernyataan (instruksi) secara berulang-ulang:
  - Hingga dicapainya satu jumlah perulangan yang ditetapkan.
  - Selama dipenuhinya kriteria yang ditetapkan.
- Dikenal juga sebagai:
  - Repetition
  - Loop
  - Iteration

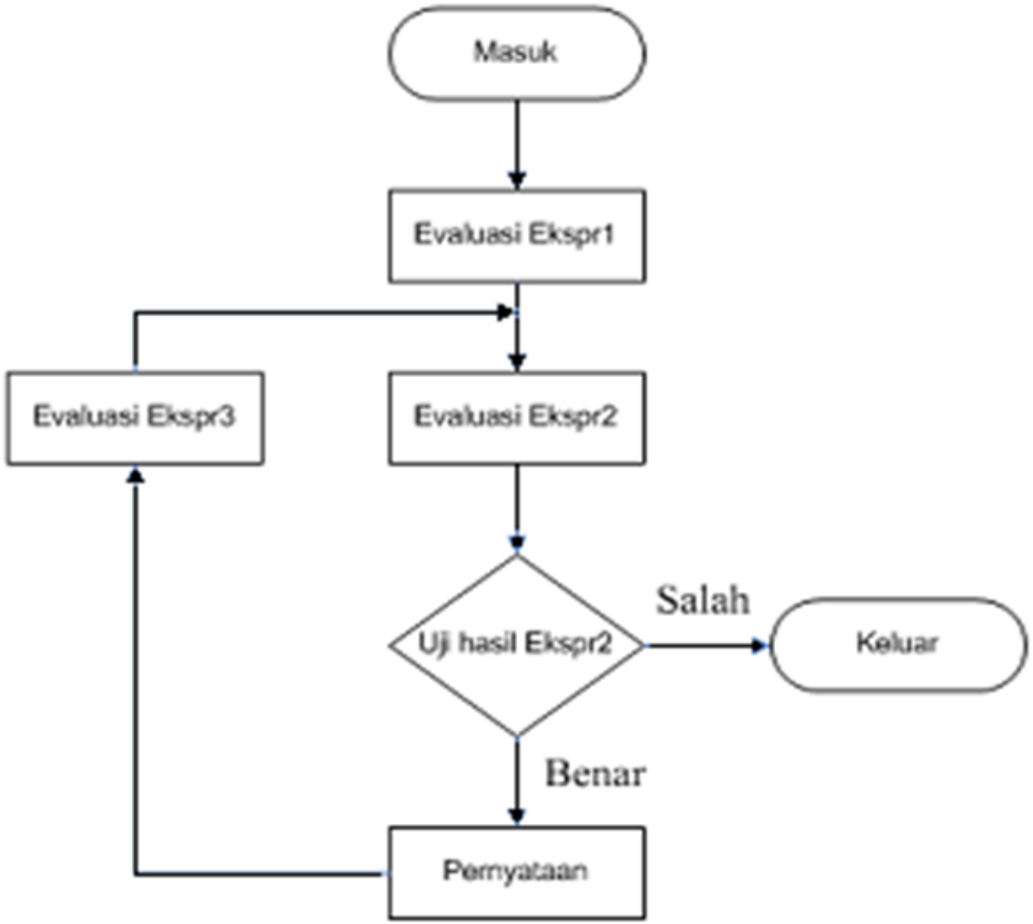
Pendahuluan

- C mengenal tiga bentuk perulangan:
  - Perulangan **for**
  - Perulangan **while**
  - Perulangan **do ... while**

Perulangan **for**

- Bentuk Umum:
  for(*ekspr1*; *ekspr2*; *ekspr3*)
  pernyataan
- Ketiga komponen perulangan **for** merupakan ekspresi-ekspresi.
- Pada umumnya;
  - *ekspr1* adalah ekspresi yang digunakan sebagai pemberi harga awal terhadap perulangan yang dilakukan;
  - *ekspr2* adalah ekspresi untuk menguji apakah perulangan masih akan terus dilakukan, dan
  - *ekspr3* adalah ekspresi untuk melakukan perubahan terhadap harga awal yang telah ditentukan.

Perulangan **for**



Gambar 1: Perulangan **for**

Perulangan **for**

- Perulangan yang berbentuk seperti ini sering disebut dengan perulangan syarat-masuk (*entry-condition loop*).
- Keputusan untuk memasuki perulangan dilakukan sebelum perulangan itu dilakukan.
- Sehingga bisa terjadi kemungkinan perulangan tidak akan pernah dilakukan.

Perulangan **for**

```
1  /* Namafile: cthfor1.c
2     Demonstrasi penggunaan perulangan for */
3  #include <stdio.h>
4  int main()
5  {
6    int i, n;
7
8    printf("Contoh perulangan for\n");
9    printf("Berikan sebuah bilangan bulat: ");
0    scanf("%d", &n);
1    for(i=0; i<n; i++)
2      printf("%d", i);
3    printf("\n");
4    return 0;
5  }
```

Perulangan **for**

```
1  /* Namafile: cthfor2.c
2     Demonstrasi penggunaan perulangan for */
3  #include <stdio.h>
4  int main()
5  {
6    int i, n;
7
8    printf("Contoh perulangan for\n");
9    printf("Berikan sebuah bilangan bulat: ");
0    scanf("%d", &n);
1    for(i=n; i>0; i--)
2      printf("%d ", i);
3    printf("\n");
4    return 0;
5  }
```

Perulangan **for**

- Perulangan **for** dalam C bersifat fleksibel.
- Misalnya pada bagian ekspresi inisialisasi, dapat diberikan lebih dari satu ekspresi yang kesemuanya hanya akan dieksekusi satu kali, yaitu pada saat perulangan **for** tersebut dimulai.

Perulangan **for**

```
1  /* Nama file: deretar.c
2     Menghitung jumlah deret aritmatika hingga suku ke
          -n, Sn = n/2 (2a + (n-1)b)) */
3  #include <stdio.h>
4  int main()
5  {
6    float a, d, sn; /* suku pertama, beda dan jlh suku
          ke-n */
7    int n, i; /* suku ke n dan pencacah */
8    printf("Berikan suku pertama deret: ");
9    scanf("%f", &a);
0    printf("Berikan beda : ");
1    scanf("%f", &d);
2    printf("Jumlah deret sampai suku ke berapa? ");
3    scanf("%d", &n);
```

Perulangan **for**

```
4    for(sn=0,i=0; i<n; i++)
5    {
6      sn += a;
7      printf("%.0f ", a);
8      a += d;
9    }
0    printf("\nJumlah deret hingga %d suku = %.2f\n", n
       , sn);
1    return 0;
2  }
```

Perulangan **for**

- Selain dgn integer, langkah perulangan juga dapat dilakukan dengan jenis data karakter.

```
1  /* Nama file : charloop.c
2     perulangan for dengan cacah karakter */
3  #include <stdio.h>
4
5  int main()
6  {
7    char ch;
8
9    for(ch = 'a'; ch <= 'z'; ch++)
0      printf("Nilai ASCII %c = %d\n", ch, ch);
1
2    return 0;
3  }
```

Perulangan **for**

- Bentuk ekspresi kedua dan ketiga perulangan **for** dapat berupa ekspresi yang lebih rumit.

```
1  /* Nama file: eksploop.c
2     perulangan for dengan ekspresi */
3  #include <stdio.h>
4  #define MAX 1024
5  int main()
6  {
7    int angka, kubik;
8
9    printf("angka\tpangkat-3\n");
0    for(angka = 2; (kubik = angka*angka*angka) < MAX;
        angka += 2)
1      printf("%d\t%d\n", angka, kubik);
2    return 0;
3  }
```

Perulangan **for**

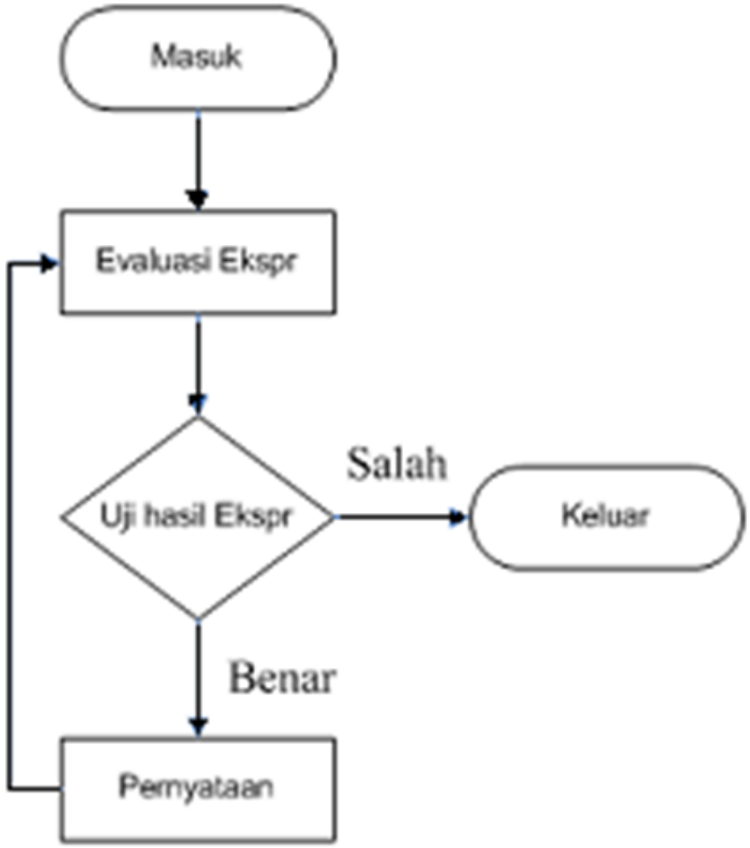
- Ekspresi pertama dapat berupa statement output sedangkan ekspresi ketiga kosong.

```
1  /* Filename: outloop.c
2     Variasi perulangan for */
3  #include <stdio.h>
4  int main()
5  {
6    int bilangan = 0;
7
8    for(printf("Berikan bilangan bulat!\n");
9      bilangan != 6; )
0      scanf("%d", &bilangan);
1    printf("Ini dia bilangan yang saya mau!\n");
2
3    return 0;
4  }
```

Perulangan **while**

- Perulangan **while** mengulang pelaksanaan pernyataan di dalam badan perulangan selama ekspresi yang diperiksa bernilai benar.
- Bentuk umum:
  while (ekspresi)
  pernyataan
- Struktur perulangan ini juga merupakan struktur perulangan syarat-masuk seperti perulangan **for**.

Perulangan **while**



Gambar 2: Perulangan **while**

Perulangan **while**

- Sebelum mengevaluasi ekspresi yang merupakan syarat masuk ke dalam perulangan, pada umumnya tetap diperlukan pemenuhan syarat awal agar perulangan dilakukan.
- Hanya saja syarat awal memasuki perulangan **while** diberikan secara terpisah dan tidak termasuk ke dalam struktur perulangan.
- Demikian pula halnya evaluasi terhadap masih dipenuhinya kondisi perulangan, yang juga diberikan secara terpisah di luar struktur perulangan.

Perulangan **while**

- Ekspresi pertama, kedua dan ketiga diletakkan secara terpisah.

```
1  /* Nama file: ujiwhile.c
2     Sifat deterministik perulangan while */
3  #include <stdio.h>
4  int main()
5  {
6    int n = 5;
7    while(n < 7)
8    {
9      printf("n = %d\n", n);
0      n++;
1      printf("Sekarang n = %d\n", n);
2    }
3    return 0;
4  }
```

Perulangan **while**

- Sifat half Boolean.

```
1  /* Nama file: halfbool.c
2       Menguji logika semi-boolean */
3  #include <stdio.h>
4
5  int main()
6  {
7    int cacah = 10;
8    while(cacah)
9      printf("%2d\n", cacah--);
0    cacah = -10;
1    while(cacah)
2      printf("%2d\n", cacah++);
3
4    return 0;
5  }
```

Perulangan **while**

- Infinite error.

```
1  /* Nama file: inferror.c
2     Kesalahan penggunaan = */
3
4  #include <stdio.h>
5
6  int main()
7  {
8    int bil, jlh = 0, status;
9    printf("Bilangan bulat untuk dijumlahkan.\n");
0    printf("Berikan s jika selesai.\n");
1    status = scanf("%d", &bil);
```

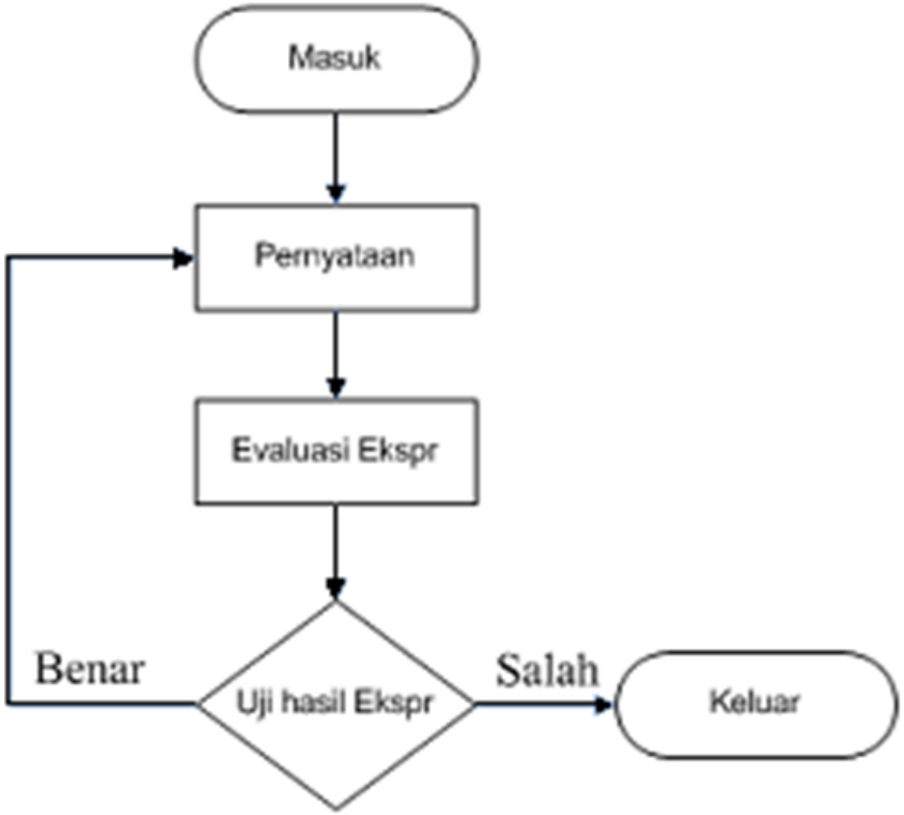
Perulangan **while**

```
2    while(status = 1)
3    {
4      jlh += bil;
5      printf("Bilangan bulat berikutnya.\n");
6      status = scanf("%d", &bil);
7    }
8    printf("jumlah bilangan = %d\n", jlh);
9
20   return 0;
21 }
```

Perulangan **do ... while**

- Pengujian terhadap ekspresi setelah selesai melaksanakan badan perulangan setiap kali selesai iterasi.
- Pengujian dilakukan di bagian akhir setelah melalui badan perulangan.
- Perulangan pastilah pernah dilaksanakan, paling sedikit satu kali.
- Dikenal sebagai perulangan syarat-keluar (*exit-condition loop*).
- Bentuk umum:
  do
  pernyataan
  while (*ekspresi*);

Perulangan **do ... while**



Gambar 3: Perulangan **do ... while**

Perulangan **do ... while**

```
1 /* Nama file: yatidak.c
2    Meminta jawaban hingga diperoleh 'y' atau 't' */
3 #include <stdio.h>
4 #include <ctype.h>
5
6 int main()
7 {
8   char jwb;
```

Perulangan **do ... while**

```
9   do
0   {
1     printf("Jawab y atau t: ");
2     jwb = getchar();
3     fflush(stdin);
4   } while (tolower(jwb) != 'y' && tolower(jwb) != 't
        ');
5   printf("Saya perlu jawaban %c saudara.\n", jwb);
6
7   return 0;
8 }
```

Perulangan **Bertingkat**

- Perulangan bertingkat (*nested loop*) adalah suatu struktur perulangan yang badan perulangannya mengandung satu atau lebih struktur perulangan yang lain.
- Struktur perulangan yang berada di bagian dalam dikenal sebagai perulangan dalam (*inner loop*), sedangkan yang berada di luar disebut perulangan luar (*outer loop*).
- Perulangan luar dan perulangan dalam itu dapat berupa struktur perulangan yang sama, dapat pula tidak.

Perulangan **Bertingkat**

```
1  /* Nama file: tblkali1.c
2     Daftar perkalian bilangan bulat 1-10 */
3  #include <stdio.h>
4  int main()
5  {
6    int kol, brs;
7    for(brs = 1; brs < 11; brs++)
8    {
9      for(kol = 1; kol < 11; kol++)
0        printf("%4d", brs * kol);
1      printf("\n");
2    }
3    return 0;
4  }
```

Pernyataan **break**, **continue**, dan **exit**

- Pernyataan **break** akan menghentikan pelaksanaan suatu perulangan, sehingga program akan dilanjutkan pada pernyataan-pernyataan yg terdapat setelah badan perulangan.
- Pernyataan **continue** akan mengembalikan kendali program ke awal perulangan dengan mengabaikan pernyataan-pernyataan yg terdapat setelah **continue**.
- Pernyataan **exit()** adalah sebuah fungsi pustaka yg memiliki sebuah argumen integer yang menyatakan status penghentian.

Pernyataan **break**, **continue**, dan **exit**

- Jika status penghentian yg diberikan adalah 0 (EXIT_SUCCESS), berarti program berhenti dengan normal.
- Nilai tidak nol (EXIT_FAILURE) menunjukkan berbagai jenis penghentian yang tidak normal.
- Fungsi ini berguna utk menghentikan eksekusi program dan mengembalikan kendali kepada sistem operasi.
- Mengembalikan nilai integer dari fungsi main() dengan pernyataan return memberikan efek yang sama seperti memanggil exit().
- Penggunaan fungsi exit() mengharuskan disertakannya berkas judul $<$ *stdlib.h* $>$.

Pernyataan **break**, **continue**, dan **exit**

- Ketiga pernyataan di atas berkaitan erat dengan pengambilan keputusan, keputusan untuk keluar dari perulangan, keputusan untuk memulai perulangan dari mula kembali, ataupun keluar dari eksekusi program.
- Oleh karena itu, ketiga pernyataan ini sering dijumpai bersama dengan pernyataan-pernyataan pengambilan keputusan (*decision*), seperti: **if**, **if-else**, **else-if** dan **switch**.

Pernyataan **break**, **continue**, dan**exit**

```
1  /* Nama file: prima.c
2      Menentukan apakah bilangan prima atau bukan */
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <math.h>
6
7  int main()
8  {
9    int i, habis = 0;
0    long x;
1
2    printf("Beri sebuah bilangan bulat,\n");
3    printf("saya akan beritahu bilangan itu ");
4    printf("prima atau bukan.\n");
5    scanf("%ld", &x);
```

Pernyataan **break**, **continue**, dan **exit**

```
16   if(x <= 1)
17   {
18     printf("Prima terkecil adalah 2.\n");
19     exit(0);
20   }
21   for(i=2; i < x; i++)
22   {
23     if((x % i) == 0)
24     {
25       printf("habis dibagi %d, karena itu",i);
26       habis = 1;
27       break;
28     }
29   }
```

Pernyataan **break**, **continue**, dan **exit**

```
30    if(!habis)
31      printf("bilangan prima.\n");
32    else
33      printf(" %ld bukan prima.\n", x);
34
35    return 0;
36  }
```

Pernyataan **break**, **continue**, dan **exit**

- Contoh program untuk menerima sembarang kalimat yang diberikan, kemudian mengubah huruf besar yang terdapat pada kalimat tersebut menjadi huruf kecil, dan sebaliknya mengubah huruf kecil menjadi huruf besar.
- Perubahan huruf tersebut sepenuhnya dilakukan dengan menggunakan operator *bitwise*.
- Pendekatannya adalah dengan memperhatikan bahwa perbedaan nomor kode ASCII antara huruf kecil dan huruf besar adalah sebesar 32 atau 2^5.

Pernyataan **break**, **continue**, dan **exit**

```
1  /* Nama file: hrfubah.c
2     Program untuk mengubah huruf kecil menjadi huruf
         besar dan sebaliknya */
3  #include <stdio.h>
4  #define MASKCHAR 32
5  int main()
6  {
7    unsigned char ch;
8    printf("Tuliskan sembarang kalimat:\n");
9    while((ch = getchar()) != '\n')
0    {
1      if((ch<65 || ch>90) && (ch<97 || ch>122))
2      {
3        printf("%c", ch);
4        continue;
5      }
```

Pernyataan **break**, **continue**, dan **exit**

```
16      printf("%c", ch^MASKCHAR);
17    }
18    printf("\n");
19
20    return 0;
21  }
```

Pernyataan **break**, **continue**, dan **exit**

- Contoh program untuk mengilustrasikan penggunaan lain kata kunci **break**, yaitu penggunaannya bersama dengan struktur kendali **switch**.
- Program ini menghitung jumlah masing-masing karakter yang dituliskan sebagai input.

Pernyataan **break**, **continue**, dan **exit**

```
1  /* Nama file: hitungk.c
2     Menghitung jumlah karakter angka, spasi dan
          karakter lain */
3  #include <stdio.h>
4
5  int main()
6  {
7    int k, nspasi, nlain2, ndigit;
8
9    nspasi = nlain2 = ndigit = 0;
0    printf("Berikan sebuah kalimat, akhiri dengan
        \'#\'\n");
1    while ((k = getchar()) != '#')
2    {
```

Pernyataan **break**, **continue**, dan **exit**

```
3    switch (k)
4    {
5      case '0':
6      case '1':
7      case '2':
8      case '3':
9      case '4':
0      case '5':
1      case '6':
2      case '7':
3      case '8':
4      case '9':
5        ndigit++;
6        break;
```

Pernyataan **break**, **continue**, dan **exit**

```
27      case '\r':
28        printf("\n");
29      case ' ' :
30      case '\t':
31        nspasi++;
32        break;
33      default  :
34        nlain2++;
35        break;
36      }
37    }
38    printf("jumlah angka=%d, ", ndigit);
39    printf("spasi=%d, lain-lain=%d.\n", nspasi, nlain2
        );
40    return 0;
41  }
```