



Pendahuluan Pengulangan

Salah satu kelebihan komputer dibandingkan dengan manusia adalah kemampuannya untuk melaksanakan suatu instruksi berulang kali tanpa mengenal lelah dan bosan. Di dalam pemrograman, perulangan dapat dilakukan sejumlah yang telah ditentukan atau sampai kondisi perulangan tercapai. Pengulangan adalah suatu proses yang melakukan perintah-perintah dalam sebuah program secara terus-menerus sampai terdapat kondisi untuk menghentikannya. Struktur pengulangan sangat membantu dalam menjaga efisiensi program.

Untuk lebih jelasnya lihat kasus berikut ini:

Tampilkan kata **"Saya tidak akan bolos lagi!"** sebanyak lima kali. Apabila tidak memahami fungsi perulangan maka seorang programmer akan membuat program seperti di bawah ini:

```
1  /*
2  Program 6.1
3  Nama File : Lat-6.1.c
4  Programmer : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     printf("Saya tidak akan bolos lagi!\n");
13     printf("Saya tidak akan bolos lagi!\n");
14     printf("Saya tidak akan bolos lagi!\n");
15     printf("Saya tidak akan bolos lagi!\n");
16     printf("Saya tidak akan bolos lagi!\n");
17     system("PAUSE");
18     return 0;
19 }
```

```
1  /*
2   Program 6.1
3   Nama File   : Lat-6.1.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     cout<<"Saya tidak akan bolos lagi!"<<endl;
15     cout<<"Saya tidak akan bolos lagi!"<<endl;
16     cout<<"Saya tidak akan bolos lagi!"<<endl;
17     cout<<"Saya tidak akan bolos lagi!"<<endl;
18     cout<<"Saya tidak akan bolos lagi!"<<endl;
19     system("PAUSE");
20     return EXIT_SUCCESS;
21 }
```

Mungkin untuk kasus seperti ini masih bisa diselesaikan dengan pemecahan di atas. Tapi apabila kata tersebut harus ditampilkan sebanyak 100 kali maka pemecahan di atas tidak efisien lagi. Struktur pengulangan dapat memecahkan kasus di atas dengan efisien.

Struktur Pengulangan

Struktur pengulangan secara umum terdiri atas dua bagian, yaitu:

1. Kondisi pengulangan

Ekspresi boolean yang harus dipenuhi untuk melaksanakan pengulangan. Kondisi ini ada yang dinyatakan secara eksplisit oleh programmer atau dikelola sendiri oleh komputer (implisit).

2. Badan pengulangan

Bagian algoritma yang diulang.

3. Inisialisasi

Aksi yang dilakukan sebelum pengulangan dilakukan pertama kali. Bagian ini akan memberi nilai awal kondisi perulangan

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

4. Terminasi

Aksi yang dilakukan setelah pengulangan selesai dilaksanakan.

Inisialisasi dan terminasi tidak harus selalu ada. Tapi kebanyakan kasus pengulangan memerlukan struktur ini. Secara umum bentuk struktur pengulangan adalah sebagai berikut:

```
<inisialisasi>  
Awal pengulangan  
    Badan pengulangan  
Akhir pengulangan  
<terminasi>
```

Di dalam pemrograman C dan C++ ada 4 bentuk pengulangan, yaitu:

1. For
2. While
3. Do while
4. Label.

Struktur Pengulangan *for*

Struktur pengulangan *for* digunakan untuk menghasilkan pengulangan sejumlah yang dispesifikasikan (sudah diketahui banyaknya). Struktur *for* mempunyai minimal pengulangan sebanyak satu kali artinya minimal dalam pengulangannya, badan pengulangan dieksekusi sebanyak satu kali. Struktur ini paling banyak digunakan untuk kasus pengulangan yang berurut dan sudah diketahui kondisi akhir pengulangannya. Untuk bisa menggunakan struktur ini, dibutuhkan sebuah variabel sebagai indeksinya.

Bentuk umum pengulangan *for* adalah sebagai berikut:

```
for(inisialisasi counter;kondisi pengulangan;statement)  
{  
    pernyataan;  
}
```

Keterangan:

1. Inisialisasi counter

Berbentuk sebuah variabel yang diinisialisasi dengan nilai kondisi awal pengulangan. Sebagai contoh: `i=1;` .

2. Kondisi pengulangan

Bagian ini berisi kondisi yang menggambarkan akhir dari pengulangan for ini. Sebagai contoh: `i<=5;`

3. Statement

Biasanya diisi dengan operasi perhitungan increment dan decrement dari variabel yang disebutkan di bagian inisialisasi counter. Apabila bagian ini dilewatkan maka pengulangan ini tidak akan berhenti (kecuali apabila di pernyataan terdapat statement increment atau decrement dari variabel tersebut).

Ada dua jenis pengulangan for, yaitu:

1. Ascending (sifatnya menaik)

Bentuk umum:

```
for(variabel=nilai_awal;kondisi;variabel++)
{
    pernyataan;
}
```

2. Descending (sifatnya menurun)

Bentuk umum:

```
for(variabel=nilai_akhir;kondisi;variabel--)
{
    pernyataan;
}
```

Sebagai contoh buatlah program untuk menampilkan angka 1 sampai 5 dan angka dari 5 sampai 1 dengan sintaks sebagai berikut:

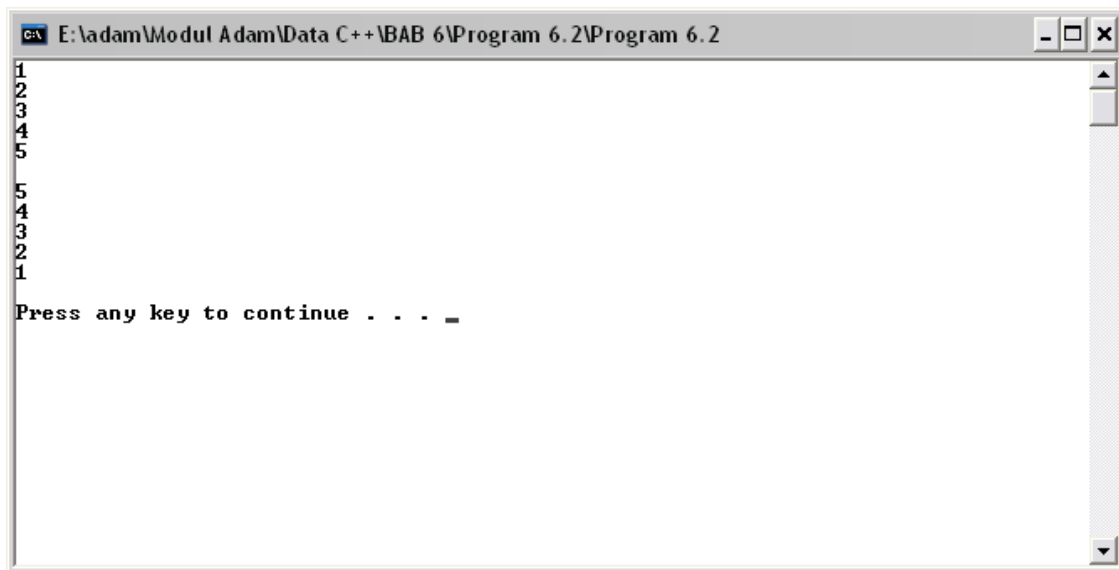
```
1  /*
2   Program 6.2
3   Nama File   : Lat-6.2.c
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int i;
13     for(i=1;i<=5;i++)
14     {
15         printf("%i\n",i);
16     }
17     printf("\n");
18     for(i=5;i>=1;i--)
19     {
20         printf("%i\n",i);
21     }
22     printf("\n");
23     system("PAUSE");
24     return 0;
25 }
```

```
1  /*
2   Program 6.2
3   Nama File   : Lat-6.2.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     int i;
15     for(i=1;i<=5;i++)
16     {
17         cout<<i<<endl;
18     }
19     cout<<endl;
20     for(i=5;i>=1;i--)
21     {
```


DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

```
22     cout<<i<<endl;
23     }
24     cout<<endl;
25     system("PAUSE");
26     return EXIT_SUCCESS;
27 }
```

Hasil eksekusi:



Gambar 6.1 Hasil eksekusi program Lat 6.2

<p>Catatan</p> 	<p>Untuk for ascending maka inialisasi counter variabelnya diisi dengan batas bawah pengulangan. Sebaliknya untuk for descending, inialisasi counter variabelnya diisi dengan batas atas pengulangannya.</p>
--	--

Salah satu pengaplikasian penggunaan for adalah untuk program penjumlahan deret.

Sintaksnya sebagai berikut:

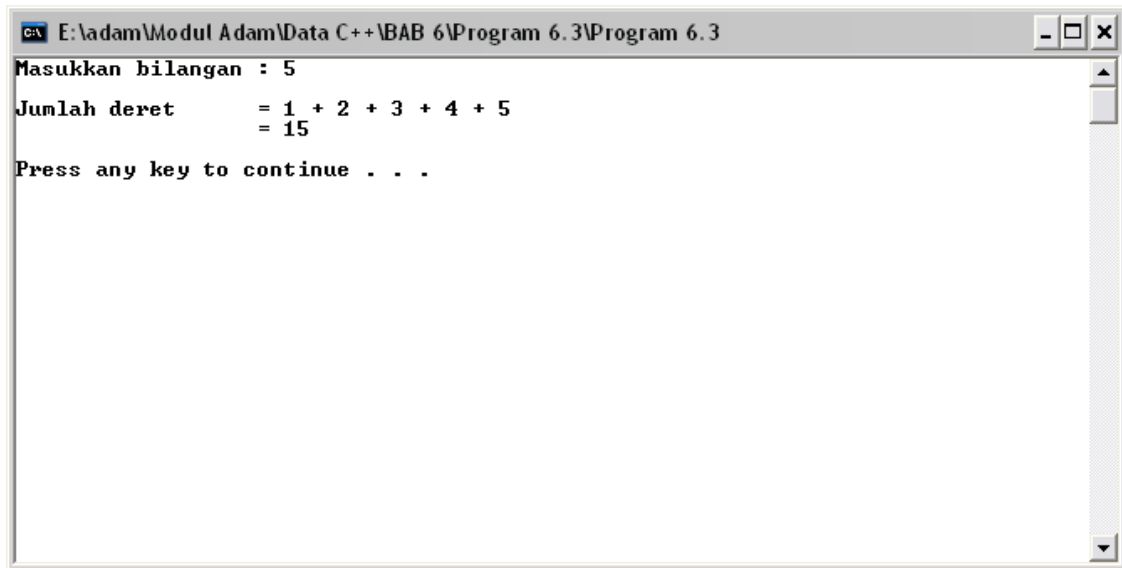
```
1  /*
2  Program 6.3
3  Nama File   : Lat-6.3.c
4  Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
```

```
11 {
12     int i,j,jumlah;
13     jumlah=0;
14     printf("Masukkan bilangan : ");scanf("%i",&j);
15     printf("\nJumlah deret      = ");
16     for(i=1;i<=j;i++)
17     {
18         jumlah=jumlah+i;
19         printf("%i ",i);
20         if(i<j)
21             printf("+ ");
22     }
23     printf("\n                = %i\n\n",jumlah);
24     system("PAUSE");
25     return 0;
26 }
```

```
1  /*
2     Program 6.3
3     Nama File   : Lat-6.3.cpp
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     int i,j,jumlah;
15     jumlah=0;
16     cout<<"Masukkan bilangan : ";cin>>j;
17     cout<<endl;
18     cout<<"Jumlah deret      = ";
19     for(i=1;i<=j;i++)
20     {
21         jumlah=jumlah+i;
22         cout<<i<<" ";
23         if(i<j)
24             cout<<"+ ";
25     }
26     cout<<endl;
27     cout<<"                = "<<jumlah<<endl<<endl;
28     system("PAUSE");
29     return EXIT_SUCCESS;
30 }
```

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

Hasil eksekusi:



```
E:\adam\Modul Adam\Data C++\BAB 6\Program 6.3\Program 6.3
Masukkan bilangan : 5
Jumlah deret      = 1 + 2 + 3 + 4 + 5
                  = 15
Press any key to continue . . .
```

Gambar 6.2 Hasil eksekusi program Lat 6.3

Struktur Pengulangan *while*

Struktur pengulangan *while* akan memeriksa kondisi pengulangan di awal blok struktur. Badan pengulangan hanya akan dilakukan apabila kondisi pengulangan bernilai *true* (benar). Oleh sebab itu, pengulangan *while* mempunyai minimal pengulangan sebanyak nol kali, artinya apabila kondisi pengulangan bernilai *false* (salah) ketika pertama kali eksekusi blok maka badan pengulangan tidak akan dieksekusi.

Bentuk umum dari pengulangan *while* adalah sebagai berikut:

```
while(kondisi)
{
    pernyataan;
}
```


DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

Untuk lebih jelasnya buatlah program di bawah ini:

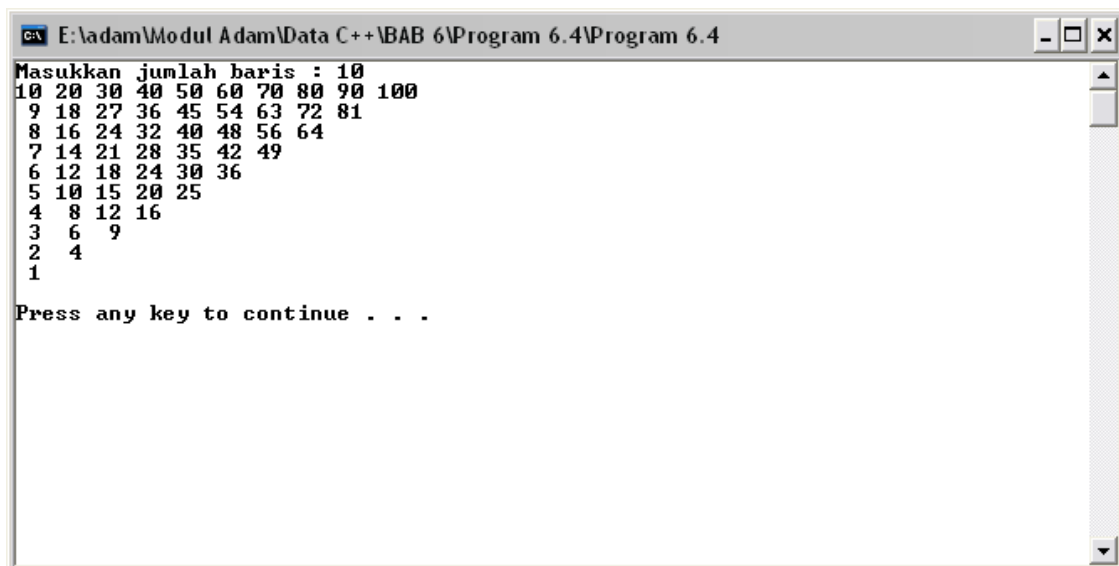
```
1  /*
2     Program 6.4
3     Nama File   : Lat-6.4.c
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int baris,kolom;
13     printf("Masukkan jumlah baris : ");scanf("%i",&baris);
14     while(baris>=1)
15     {
16         kolom=1;
17         while(kolom<=baris)
18         {
19             printf("%2i ",baris*kolom);
20             kolom++;
21         }
22         printf("\n");
23         baris--;
24     }
25     printf("\n");
26     system("PAUSE");
27     return 0;
28 }
```

```
1  /*
2     Program 6.4
3     Nama File   : Lat-6.4.cpp
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9  #include <iomanip>
10
11 using namespace std;
12
13 int main(int argc, char *argv[])
14 {
15     int baris,kolom;
16     cout<<"Masukkan jumlah baris : ";cin>>baris;
```

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

```
17     while(baris>=1)
18     {
19         kolom=1;
20         while(kolom<=baris)
21         {
22             cout<<setw(2)<<baris*kolom<<" ";
23             kolom++;
24         }
25         cout<<endl;
26         baris--;
27     }
28     cout<<endl;
29     system("PAUSE");
30     return EXIT_SUCCESS;
31 }
```

Hasil eksekusi:



Gambar 6.3 Hasil eksekusi program Lat 6.4

Salah satu pengaplikasian pengulangan while adalah untuk menghitung rata-rata sampai kondisi yang tidak diinputkan di awal (contohnya sampai nilai masih memenuhi ketentuan umum yaitu antara 0 sampai 100). Sintaksnya adalah sebagai berikut:

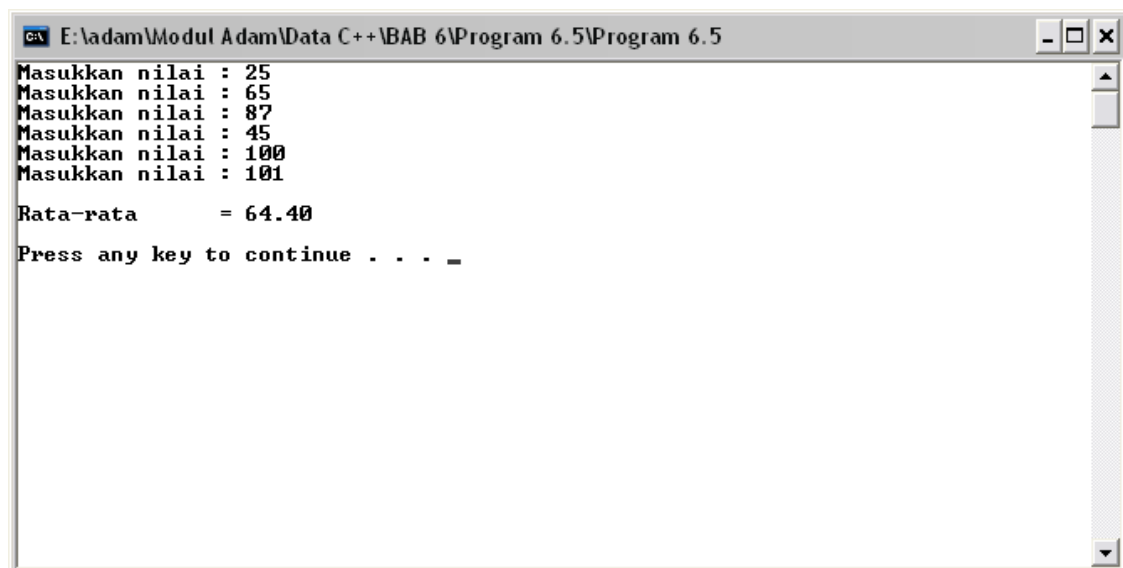
```
1  /*
2  Program 6.5
3  Nama File   : Lat-6.5.c
4  Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
```

```
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int jml_data;
13     float nilai,rata2,jml_nilai;
14     jml_data=0;
15     jml_nilai=0;
16     printf("Masukkan nilai : ");scanf("%f",&nilai);
17     while((nilai>=0)&&(nilai<=100))
18     {
19         jml_data=jml_data+1;
20         jml_nilai=jml_nilai+nilai;
21         printf("Masukkan nilai : ");scanf("%f",&nilai);
22     }
23     if(jml_data!=0)
24     {
25         rata2=jml_nilai/jml_data;
26         printf("\nRata-rata      = %.2f\n\n",rata2);
27     }
28     else
29         printf("\nTidak ada data inputan!\n\n");
30     system("PAUSE");
31     return 0;
32 }
```

```
1  /*
2     Program 6.5
3     Nama File   : Lat-6.5.cpp
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9  #include <iomanip>
10
11 using namespace std;
12
13 int main(int argc, char *argv[])
14 {
15     int jml_data;
16     float nilai,rata2,jml_nilai;
17     jml_data=0;
18     jml_nilai=0;
19     cout<<"Masukkan nilai : ";cin>>nilai;
20     while((nilai>=0)&&(nilai<=100))
```

```
21     {
22         jml_data=jml_data+1;
23         jml_nilai=jml_nilai+nilai;
24         cout<<"Masukkan nilai : ";cin>>nilai;
25     }
26     if(jml_data!=0)
27     {
28         rata2=jml_nilai/jml_data;
29         cout<<endl;
30         cout<<setiosflags(ios::fixed);
31         cout<<"Rata-rata      = "<<setprecision(2)
32             <<rata2<<endl<<endl;
33     }
34     else
35     {
36         cout<<endl;
37         cout<<"Tidak ada data inputan!"<<endl<<endl;
38     }
39     system("PAUSE");
40     return EXIT_SUCCESS;
}
```

Hasil eksekusi:



Gambar 6.4 Hasil eksekusi program Lat 6.4

Struktur Pengulangan *do while*

Pengulangan *do while* akan memeriksa kondisi pengulangan di akhir blok pengulangan. Hal ini tentunya sangat berbeda dengan pengulangan *while* yang memeriksa kondisi pengulangan di awal blok. Perintah ini ekuivalen dengan perintah **repeat until** yang ada di algoritma. Karena pemeriksaan kondisi ada di akhir blok maka badan pengulangan minimal akan dieksekusi sebanyak satu kali.

Bentuk umum pengulangan *do while* adalah sebagai berikut:

```
do
{
    pernyataan;
}while(kondisi);
```

Ada beberapa pengaplikasian pengulangan *do while* antara lain untuk membuat struktur menu (beserta validasi inputannya) serta untuk membuat login menggunakan password (kata kunci). Untuk pengaplikasian pada struktur menu, programnya adalah sebagai berikut:

```
1  /*
2  Program 6.6
3  Nama File : Lat-6.6.c
4  Programmer : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <conio2.h>
10
11 int main(int argc, char *argv[])
12 {
13     int pil;
14     int i;
15     do
16     {
17         printf(" =====\n");
18         printf(" |           Menu           |\n");
19         printf(" =====\n");
20         printf(" | 1. Tampilkan Hai         |\n");
21         printf(" | 2. Tampilkan Hoi         |\n");
22         printf(" | 3. Tampilkan Hui         |\n");
```

```

23     printf("| 4. Keluar          |\n");
24     printf("=====\n");
25     printf("| Pilihan [1..4] :      |\n");
26     printf("=====\n");
27     gotoxy(20,9);scanf("%d",&pil);
28     gotoxy(3,12);
29     switch(pil)
30     {
31         case 1 :printf("Hai\n\n");break;
32         case 2 :printf("Hoi\n\n");break;
33         case 3 :printf("Hui\n\n");break;
34         case 4 :printf("Bye\n\n");break;
35         default:printf("Anda salah pilih\n\n");
36     }
37     system("PAUSE");
38     system("cls");
39 }while(pil!=4);
40 printf("\n\n");
41 return 0;
42 }

```

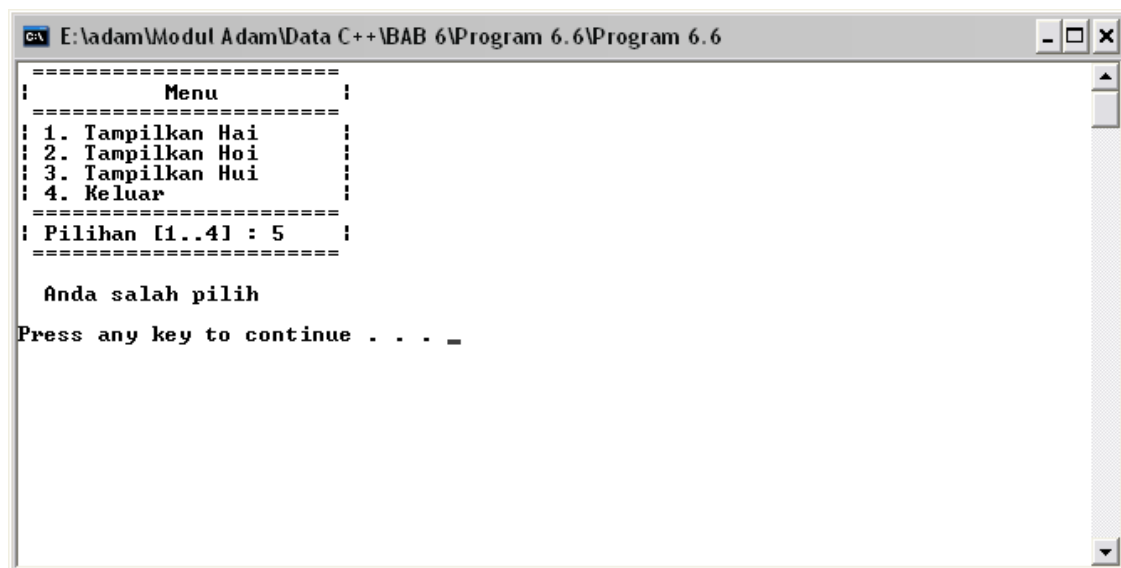
```

1  /*
2   Program 6.6
3   Nama File   : Lat-6.6.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9  #include <conio2.h>
10
11 using namespace std;
12
13 int main(int argc, char *argv[])
14 {
15     int pil;
16     int i;
17     do
18     {
19         cout<<"====="<<endl;
20         cout<<"|          Menu          |"<<endl;
21         cout<<"====="<<endl;
22         cout<<"| 1. Tampilkan Hai      |"<<endl;
23         cout<<"| 2. Tampilkan Hoi      |"<<endl;
24         cout<<"| 3. Tampilkan Hui      |"<<endl;
25         cout<<"| 4. Keluar             |"<<endl;
26         cout<<"====="<<endl;

```

```
27     cout<<" | Pilihan [1..4] :      |"<<endl;
28     cout<<" =====<<endl;
29     gotoxy(20,9);cin>>pil;
30     gotoxy(3,12);
31     switch(pil)
32     {
33         case 1 :cout<<"Hai"<<endl<<endl;break;
34         case 2 :cout<<"Hoi"<<endl<<endl;break;
35         case 3 :cout<<"Hui"<<endl<<endl;break;
36         case 4 :cout<<"Bye"<<endl<<endl;break;
37         default:cout<<"Anda salah pilih"
38                 <<endl<<endl;
39     }
40     system("PAUSE");
41     system("cls");
42     }while(pil!=4);
43     return EXIT_SUCCESS;
44 }
```

Hasil eksekusi:



Gambar 6.5 Hasil eksekusi program Lat 6.6

Selain itu pengaplikasian do while yang kedua adalah untuk membuat login yang akan membatasi password yang salah sebanyak tiga kali. Sintaksnya adalah sebagai berikut:

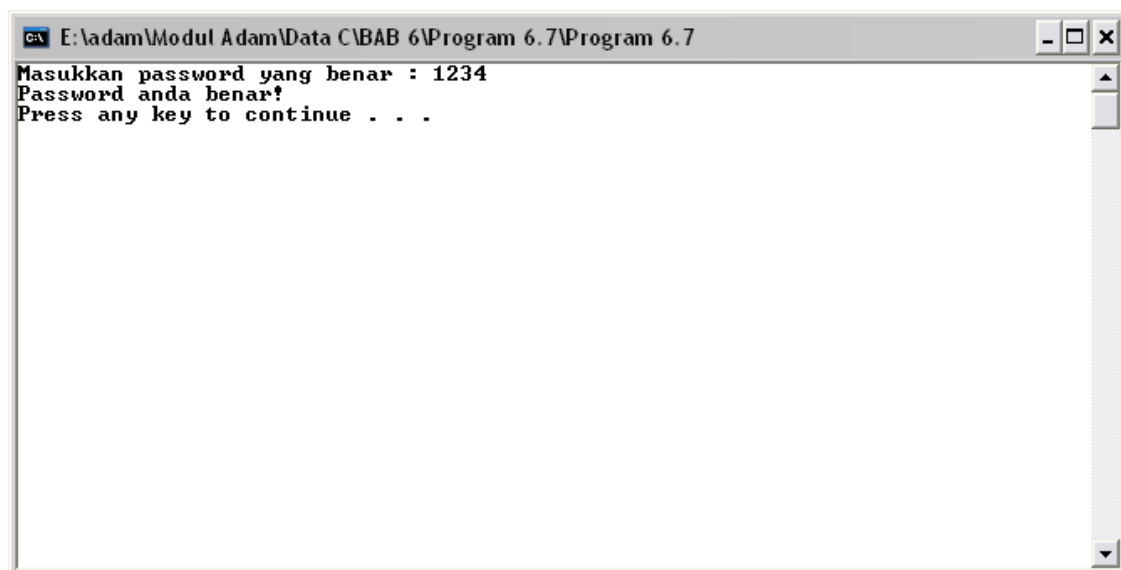
```
1  /*
2   Program 6.7
3   Nama File   : Lat-6.7.c
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <conio.h>
10
11 #define pass 1234
12 int main(int argc, char *argv[])
13 {
14     int password,i;
15     i=0;
16     do
17     {
18         printf("Masukkan password yang benar : ");
19         scanf("%i",&password);
20         if(password==pass)
21             printf("Password anda benar!\n");
22         else
23         {
24             printf("Password anda salah\n");
25             getch();
26             system("cls");
27         }
28         i++;
29     }while((password!=pass)&&(i!=3));
30     system("PAUSE");
31     return 0;
32 }
```

```
1  /*
2   Program 6.7
3   Nama File   : Lat-6.7.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9  #include <conio.h>
10
11 #define pass 1234
12 using namespace std;
13
14 int main(int argc, char *argv[])
```



```
15 {
16     int password,i;
17     i=0;
18     do
19     {
20         cout<<"Masukkan password yang benar : ";
21         cin>>password;
22         if(password==pass)
23             cout<<"Password anda benar!"<<endl;
24         else
25         {
26             cout<<"Password anda salah"<<endl;
27             getch();
28             system("cls");
29         }
30         i++;
31     }while((password!=pass)&&(i!=3));
32     system("PAUSE");
33     return EXIT_SUCCESS;
34 }
```

Hasil eksekusi:



Gambar 6.6 Hasil eksekusi program Lat 6.7

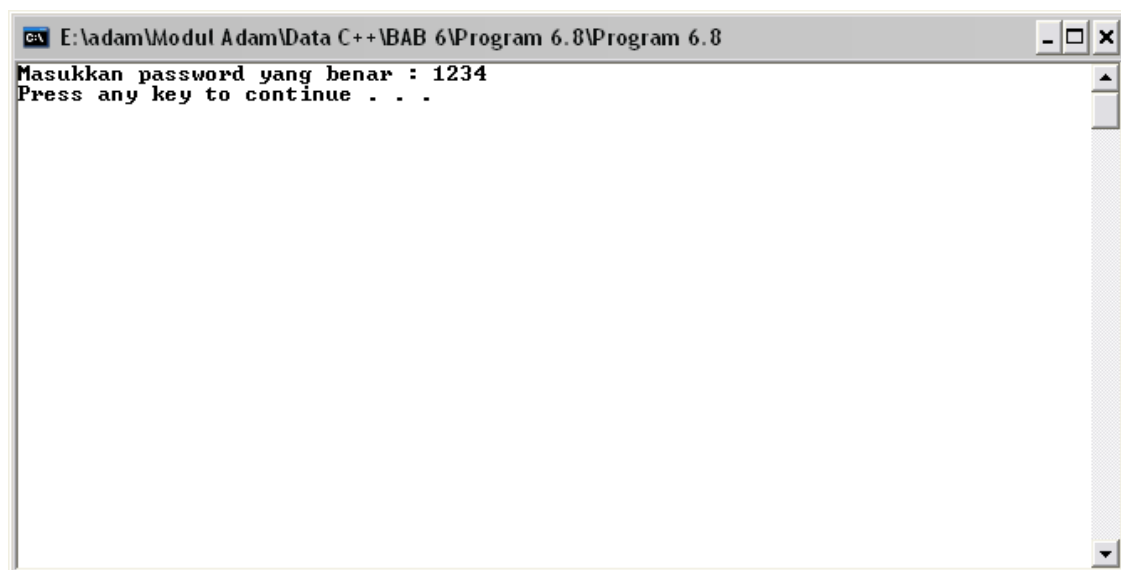
Selain bisa menggunakan do while, program password juga bisa dibangun menggunakan struktur while. Sintaksnya adalah sebagai berikut:

```
1  /*
2   Program 6.8
3   Nama File   : Lat-6.8.c
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <conio2.h>
10
11 #define pass 1234
12 int main(int argc, char *argv[])
13 {
14     int password,i;
15     i=1;
16     printf("Masukkan password yang benar : ");
17     scanf("%i",&password);
18     while((password!=pass)&&(i!=3))
19     {
20         printf("Password anda salah!\n\n");
21         getch();
22         gotoxy(1,2);clrhol();//menghapus satu baris
23                                 dimulai dari pointer
24         gotoxy(32,1);clrhol();
25         scanf("%i",&password);
26         i++;
27     }
28     system("PAUSE");
29     return 0;
30 }
```

```
1  /*
2   Program 6.8
3   Nama File   : Lat-6.8.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9  #include <conio2.h>
10
11 #define pass 1234
12 using namespace std;
13
14 int main(int argc, char *argv[])
15 {
16     int password,i;
```

```
17     i=1;
18     cout<<"Masukkan password yang benar : ";
19     cin>>password;
20     while((password!=pass)&&(i!=3))
21     {
22         cout<<"Password anda salah!"<<endl<<endl;
23         getch();
24         gotoxy(1,2);clrhol();//menghapus satu baris
25                                     dimulai dari pointer
26         gotoxy(32,1);clrhol();
27         scanf("%i",&password);
28         i++;
29     }
30     system("PAUSE");
31     return EXIT_SUCCESS;
32 }
```

Hasil eksekusi:



```
E:\adam\Modul Adam\Data C++\BAB 6\Program 6.8\Program 6.8
Masukkan password yang benar : 1234
Press any key to continue . . .
```

Gambar 6.7 Hasil eksekusi program Lat 6.8

Struktur Pengulangan Menggunakan Label

Pengulangan dengan teknik label adalah teknik pengulangan yang pertama kali dikenal (bahasa assembly). Penggunaan label mempunyai banyak kelebihan akan tetapi sangat tidak dianjurkan penggunaannya untuk memecahkan suatu kasus pengulangan. Label akan membuat alur program yang struktural menjadi berantakan. Ada dua kasus yang menggambarkan kelemahan label, yaitu:

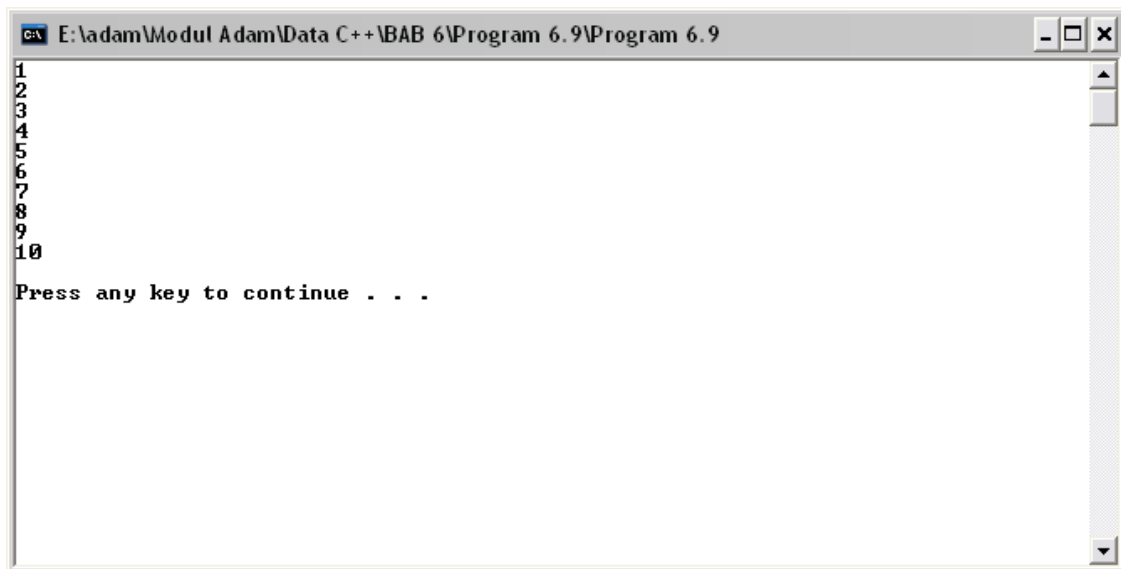
1. Kasus baris yang tidak tereksekusi

```
1  /*
2     Program 6.9
3     Nama File   : Lat-6.9.c
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int i;
13     i=0;
14     awal:
15         i=i+1;
16         printf("%i\n",i);
17         if(i<10)
18             goto awal;
19         else
20             goto akhir;
21         printf("Perintah ini tak akan dieksekusi\n");
22         printf("Perintah ini juga tak akan
23             dieksekusi\n");
24     akhir:
25         printf("\n");
26         system("PAUSE");
27         return 0;
28 }
```

```
1  /*
2     Program 6.9
3     Nama File   : Lat-6.9.cpp
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     int i;
15     i=0;
```


```
16     awal:
17         i=i+1;
18         cout<<i<<endl;
19         if(i<10)
20             goto awal;
21         else
22             goto akhir;
23         cout<<"Perintah ini tak akan dieksekusi"<<endl;
24         cout<<"Perintah ini juga tak akan dieksekusi"
                <<endl;
25
26     akhir:
27         cout<<endl;
28         system("PAUSE");
29         return EXIT_SUCCESS;
30 }
```

Hasil eksekusi:



Gambar 6.8 Hasil eksekusi program Lat 6.9

Terlihat dari hasil eksekusi bahwa ada dua baris perintah yang tidak dieksekusi sama sekali diakibatkan adanya label. Hal ini jelas membuat struktur program menjadi tidak baik.

<p>Catatan</p> 	<p>Untuk beberapa kasus, pengulangan menggunakan label sangat membantu. Tetapi apabila tidak hati-hati dalam peletakan labelnya akan mengakibatkan adanya baris yang tidak tereksekusi dan membuat alur program menjadi rumit.</p>
--	--

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

2. Kasus label yang mengingkari kondisi if (merusak logika program)

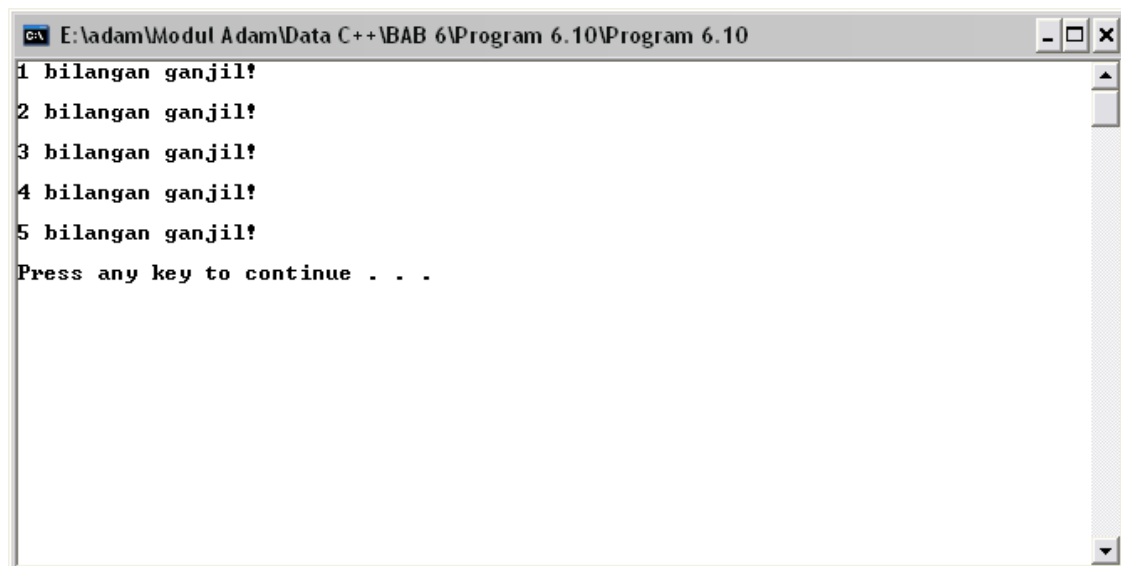
```
1  /*
2     Program 6.10
3     Nama File   : Lat-6.10.c
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int i;
13     for(i=1;i<=5;i++)
14     {
15         goto free_pass;
16         if(i%2==0)
17             printf("%i bilangan genap!\n\n",i);
18         else
19         {
20             free_pass:
21                 printf("%i bilangan ganjil!\n\n",i);
22         }
23     }
24     system("PAUSE");
25     return 0;
26 }
```

```
1  /*
2     Program 6.10
3     Nama File   : Lat-6.10.cpp
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     int i;
15     for(i=1;i<=5;i++)
16     {
17         goto free_pass;
18         if(i%2==0)
```

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

```
19         cout<<i<<" bilangan genap!"<<endl<<endl;  
20     else  
21     {  
22         free_pass:  
23         cout<<i<<" bilangan ganjil!"  
                <<endl<<endl;  
24     }  
25 }  
26 system("PAUSE");  
27 return EXIT_SUCCESS;  
28 }
```

Hasil eksekusi:



```
E:\adam\Modul Adam\Data C++\BAB 6\Program 6.10\Program 6.10  
1 bilangan ganjil!  
2 bilangan ganjil!  
3 bilangan ganjil!  
4 bilangan ganjil!  
5 bilangan ganjil!  
Press any key to continue . . .
```

Gambar 6.9 Hasil eksekusi program Lat 6.10

Dari hasil eksekusi terlihat bahwa label membuat logika program menjadi salah. Angka 2 dan 4 yang seharusnya masuk ke dalam kategori bilangan genap malah ditampilkan sebagai bilangan ganjil. Hal itu disebabkan karena adanya label `free_pass` yang langsung masuk ke dalam `else`.

Perintah *break* dan *continue*

Perintah `break` digunakan untuk menghentikan eksekusi pengulangan dan program akan langsung meloncat ke pernyataan yang ada di blok selanjutnya. Untuk dapat memahami penggunaan perintah `break` lihat program berikut ini:

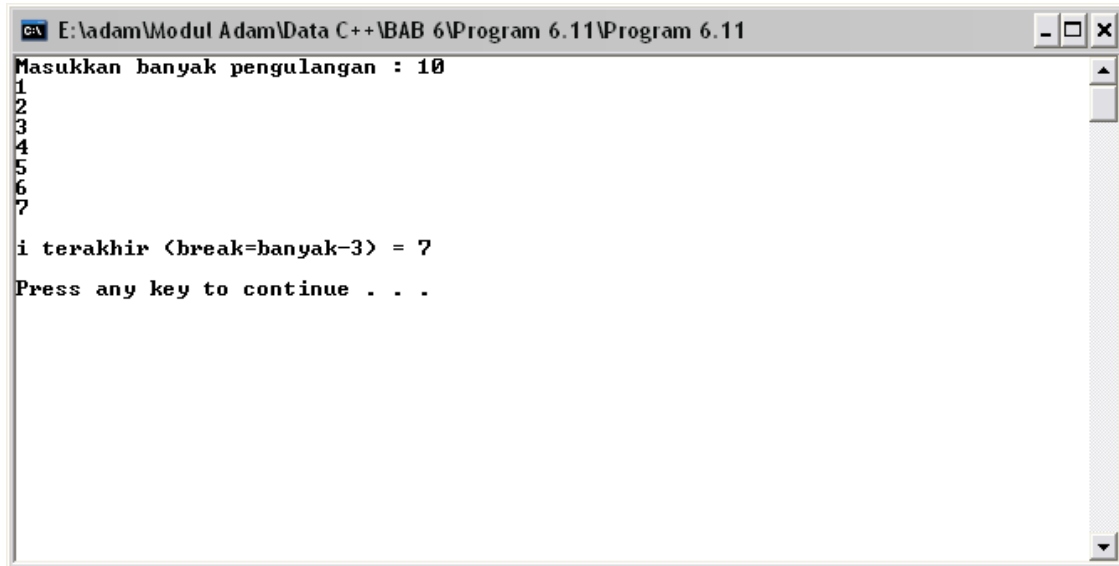
```
1  /*
2   Program 6.11
3   Nama File   : Lat-6.11.c
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int i,j;
13     printf("Masukkan banyak pengulangan : ");
14     scanf("%i",&j);
15     for(i=1;i<=j;i++)
16     {
17         printf("%i\n",i);
18         if(i==(j-3))
19             break;
20     }
21     printf("\ni terakhir (break=banyak-3) = %i\n\n",i);
22     system("PAUSE");
23     return 0;
24 }
```

```
1  /*
2   Program 6.11
3   Nama File   : Lat-6.11.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     int i,j;
15     cout<<"Masukkan banyak pengulangan : ";cin>>j;
16     for(i=1;i<=j;i++)
17     {
18         cout<<i<<endl;
19         if(i==(j-3))
20             break;
21     }
22     cout<<endl;
```


DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

```
23     cout<<"i terakhir (break=banyak-3) = "<<i
      <<endl<<endl;
24     system("PAUSE");
25     return EXIT_SUCCESS;
26 }
```

Hasil eksekusi:



```
E:\adam\Modul Adam\Data C++\BAB 6\Program 6.11\Program 6.11
Masukkan banyak pengulangan : 10
1
2
3
4
5
6
7
i terakhir (break=banyak-3) = 7
Press any key to continue . . .
```

Gambar 6.10 Hasil eksekusi program Lat 6.11

Dari hasil eksekusi terlihat bahwa pengulangan akan dihentikan apabila *i*-nya sudah sama dengan banyak pengulangan dikurangi tiga.

Berbeda dengan perintah `break`, perintah `continue` berguna untuk meneruskan perulangan. Dengan kata lain perintah ini akan menyebabkan program meloncat ke pernyataan awal yang terdapat di dalam badan pengulangan. Untuk lebih jelasnya lihat program di bawah ini:

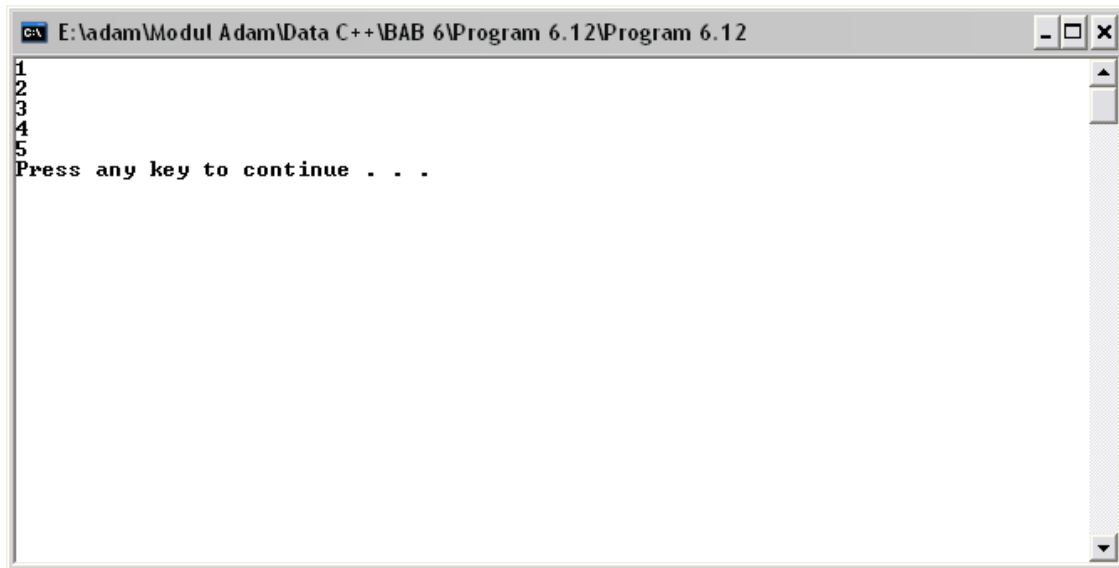
```
1  /*
2  Program 6.12
3  Nama File   : Lat-6.12.c
4  Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main(int argc, char *argv[])
11 {
12     int bil;
```

```
13     for(bil=1;bil<=5;bil++)
14     {
15         printf("%i\n",bil);
16         continue;
17         printf("Diskriminasi!!!! Saya kok tidak
18             muncul??");
19     }
20     system("PAUSE");
21     return 0;
22 }
```

```
1  /*
2     Program 6.12
3     Nama File   : Lat-6.12.c
4     Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <cstdlib>
8  #include <iostream>
9
10 using namespace std;
11
12 int main(int argc, char *argv[])
13 {
14     int bil;
15     for(bil=1;bil<=5;bil++)
16     {
17         cout<<bil<<endl;
18         continue;
19         cout<<"Diskriminasi!!!! Saya kok tidak
20             muncul??";
21     }
22     system("PAUSE");
23     return EXIT_SUCCESS;
24 }
```

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

Hasil eksekusi:



```
E:\adam\Modul Adam\Data C++\BAB 6\Program 6.12\Program 6.12
1
2
3
4
5
Press any key to continue . . .
```

Gambar 6.11 Hasil eksekusi program Lat 6.12

Dari hasil eksekusi terlihat bahwa ada satu baris perintah di dalam badan pengulangan yang tidak dieksekusi akibat adanya perintah continue sebelumnya. Perintah continue ini akan menyebabkan eksekusi pengulangan selanjutnya tanpa membereskan terlebih dahulu badan pengulangan yang terletak di bawah perintah continue.

Kombinasi Pengulangan dengan Conio2.h

Salah satu fungsi pengulangan yang bisa digunakan adalah untuk membentuk kombinasi warna dalam satu layar. Tentunya untuk dapat menggunakan warna di dev c++ harus menginstal conio2.h seperti yang sudah disebutkan di bab IV. Salah satu contoh programnya adalah sebagai berikut:

```
1  /*
2   Program 6.13
3   Nama File   : Lat-6.13.c
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <conio2.h>
10
```

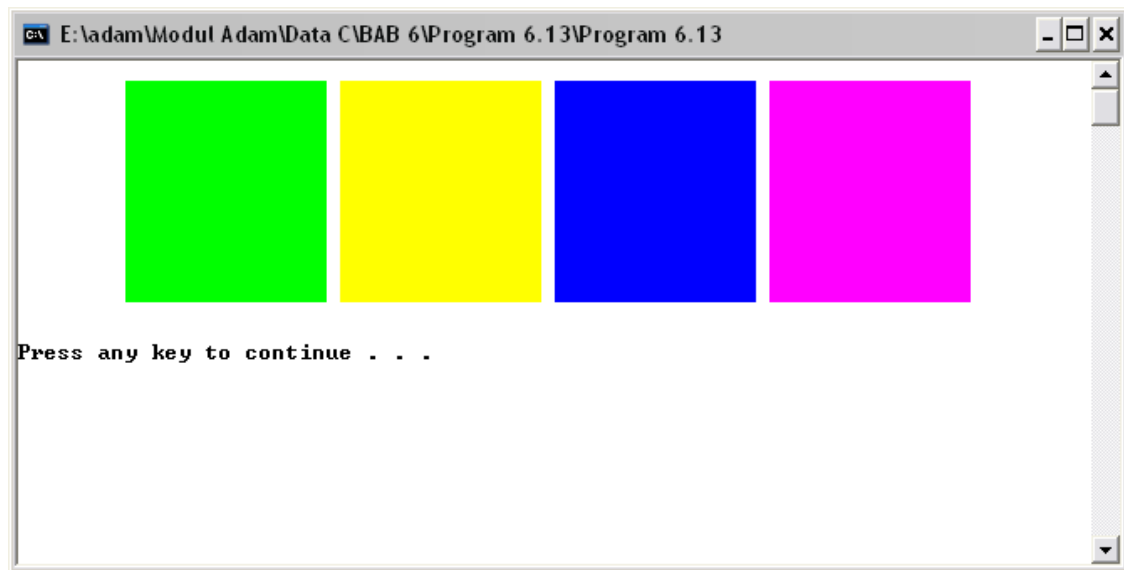
```
11 int main(int argc, char *argv[])
12 {
13     int i;
14     textbackground(WHITE);system("cls");
15     for(i=2;i<=12;i++)
16     {
17         gotoxy(9,i);
18         textbackground(LIGHTGREEN);
19         printf("                ");
20     }
21
22     for(i=2;i<=12;i++)
23     {
24         gotoxy(25,i);
25         textbackground(YELLOW);
26         printf("                ");
27     }
28
29     for(i=2;i<=12;i++)
30     {
31         gotoxy(41,i);
32         textbackground(LIGHTBLUE);
33         printf("                ");
34     }
35
36     for(i=2;i<=12;i++)
37     {
38         gotoxy(57,i);
39         textbackground(LIGHTMAGENTA);
40         printf("                ");
41     }
42     textbackground(WHITE);
43     printf("\n\n\n");
44     textcolor(BLACK);
45     system("PAUSE");
46     return 0;
47 }
```

```
1  /*
2   Program 6.13
3   Nama File   : Lat-6.13.cpp
4   Programmer  : Adam Mukharil Bachtiar, S.Kom.
5   */
6
7  #include <cstdlib>
8  #include <iostream>
9  #include <conio2.h>
```

```
10
11 using namespace std;
12
13 int main(int argc, char *argv[])
14 {
15     int i;
16     for(i=2;i<=12;i++)
17     {
18         gotoxy(9,i);
19         textbackground(LIGHTGREEN);
20         cout<<"          ";
21     }
22
23     for(i=2;i<=12;i++)
24     {
25         gotoxy(25,i);
26         textbackground(YELLOW);
27         cout<<"          ";
28     }
29
30     for(i=2;i<=12;i++)
31     {
32         gotoxy(41,i);
33         textbackground(LIGHTBLUE);
34         cout<<"          ";
35     }
36
37     for(i=2;i<=12;i++)
38     {
39         gotoxy(57,i);
40         textbackground(LIGHTMAGENTA);
41         cout<<"          ";
42     }
43     textbackground(WHITE);
44     cout<<endl<<endl<<endl;
45     textcolor(BLACK);
46     system("PAUSE");
47     return EXIT_SUCCESS;
48 }
```

DISUSUN OLEH : ADAM MUKHARIL BACHTIAR, S.Kom.

Hasil eksekusi:



Gambar 6.12 Hasil eksekusi program Lat 6.13