

Lembar Kerja
Praktikum Pemrograman Dasar



Nama Praktikan	Adi Febriana	NPM 140910200022
Tanggal Praktikum	12 November 2020	
Anggota Kelompok	Lidia Marserlina	NPM 140910200021
	Adi Febriana	NPM 140910200022
	Fata Hibrizi Thufail Wijdan	NPM 140910200023
	Frizqi Ramadhandika Listanto	NPM 140910200024
	Athaya Salsabil	NPM 140910200025
Asisten	Rafly	
Modul	Modul 3 Fungsi (3.3.1, 3.3.5, 3.3.8, 3.3.10, 3.3.13)	

PROGRAM STUDI TEKNIK ELEKTRO
UNIVERSITAS PADJADJARAN

2020

LEMBAR PENGESAHAN

Nama Praktikan	Adi Febriana	NPM 140910200022
Tanggal Praktikum	12 November 2020	
Asisten	Rafly	
Modul	Modul 3 Fungsi (3.3.1, 3.3.5, 3.3.8, 3.3.10, 3.3.13)	
Tujuan	Setelah mengikuti praktikum dengan pokok bahasan fungsi mahasiswa akan dapat memecahkan program besar yang kompleks menjadi program-program modular yang lebih kecil dengan benar	

Nilai	Asisten

Modul 2

Pokok Bahasan

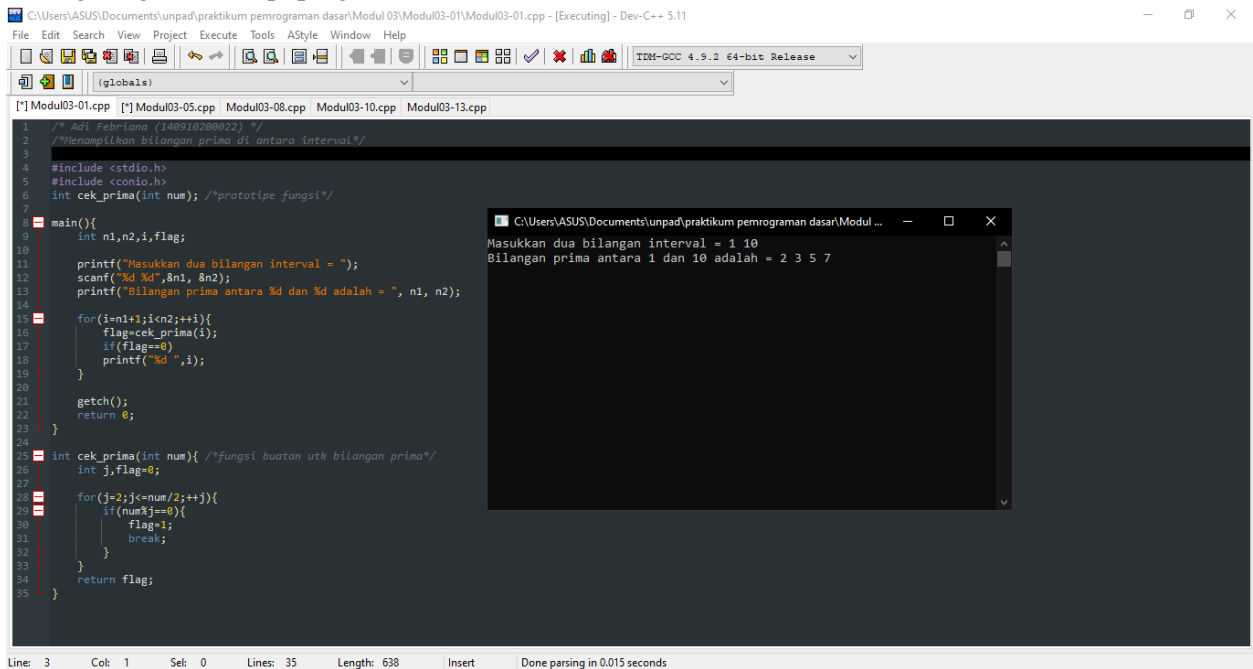
1. Pengenalan fungsi
2. Fungsi buatan
3. Tipe fungsi
4. Rekursi
5. Kelas penyimpanan (*Storage Class*)

Tujuan

Setelah mengikuti praktikum dengan pokok bahasan fungsi mahasiswa akan dapat memecahkan program besar yang kompleks menjadi program-program modular yang lebih kecil dengan benar

Percobaan

1.1 Listing Program / Script program



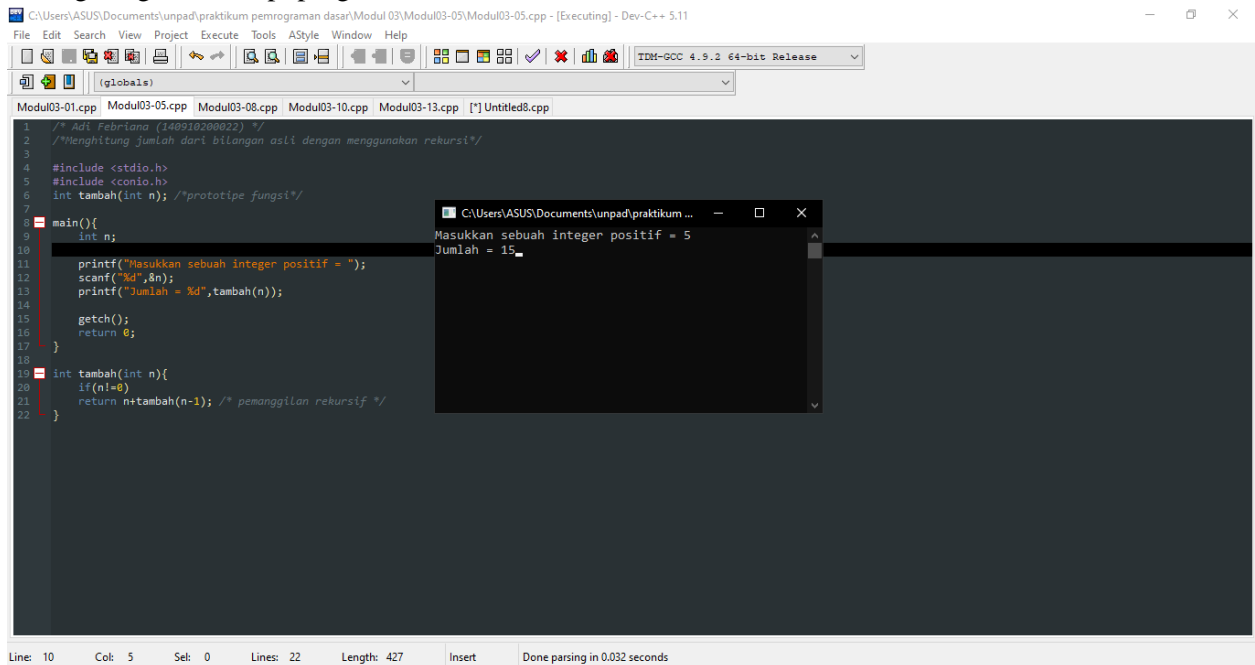
```
1  /* Adi Febrina (140910200022) */
2  //Menampilkan bilangan prima di antara interval*/
3
4  #include <stdio.h>
5  #include <conio.h>
6  int cek_prima(int num); /*prototipe fungsi*/
7
8  main(){
9      int n1,n2,i,flag;
10
11      printf("Masukkan dua bilangan interval = ");
12      scanf("%d %d",&n1, &n2);
13      printf("Bilangan prima antara %d dan %d adalah = ", n1, n2);
14
15      for(i=n1+1;i<n2;++i){
16          flag=cek_prima(i);
17          if(flag==0)
18              printf("%d ",i);
19      }
20
21      getch();
22      return 0;
23  }
24
25  int cek_prima(int num){ /*fungsi buatan utk bilangan prima*/
26      int j,flag=0;
27
28      for(j=2;j<=num/2;++j){
29          if(num%j==0){
30              flag=1;
31              break;
32          }
33      }
34      return flag;
35  }
```

Masukkan dua bilangan interval = 1 10
Bilangan prima antara 1 dan 10 adalah = 2 3 5 7

Analisis :

Pada baris 6 prototipe fungsi cek_prima di deklarasi, pada baris 9 dideklarasikan variabel n1, n2, i, flag dengan tipe data integer. Pada baris 12, data 2 bilangan interval diinput ke variabel n1 dan n2. Pada baris 15 terdapat sebuah pengulangan dimulai dengan variabel i = variabel n1 + 1, selama variabel i kurang dari variabel n2, pengulangan tetap berlanjut, dan setiap akhir pengulangan variabel I ditambah 1. Pada baris 16, variabel flag berisi nilai dari fungsi cek_prima dengan parameter variabel i, pada baris 17 ada pengecekan variabel flag, ketika variabel flag = 0 maka akan tampil variabel i pada konsol. Pada fungsi cek_prima, di deklarasi variabel j dan flag (dengan nilai 0) dengan tipe data integer, pada baris 17 terjadi pengulangan dimulai dengan variabel j = 2, selama variabel j kurang dari variabel i/2 maka pengulangan tetap berlanjut, dan setiap akhir pengulangan variabel j ditambah j. pada baris 28, terjadi pengecekan, ketika variabel i / variabel j sisanya kurang 0, maka flag akan bernilai 1 dan pengulangan berakhir. Pada kasus gambar diatas, interval yang dimasukkan adalah 1 sampai 5. Pada baris 15, pengulangan dimulai dengan variabel i bernilai 2, selama 2 kurang dari 5 maka pengulangan akan berlanjut, dan setiap akhir pengulangan variabel i ditambah 1. Pada baris 16, fungsi cek_prima dengan parameter 2. Lompat pada baris 27, pengulangan dimulai dengan variabel j = 2, selama 2 kurang dari 1/2 maka pengulangan akan berlanjut, dan setiap pengulangan variabel j akan ditambah 1. Karena $2 \leq \frac{1}{2}$ maka pengulangan berakhir, dan flag akan bernilai 0. Pada baris 17, pengecekan terjadi dan terpenuhi karena flag = 0, maka variabel I yang dimana nilainya 2 akan muncul di konsol. Dan pengulangan baris 15 akan terus berlanjut hingga variabel i > 5.

1.2 Listing Program / Script program



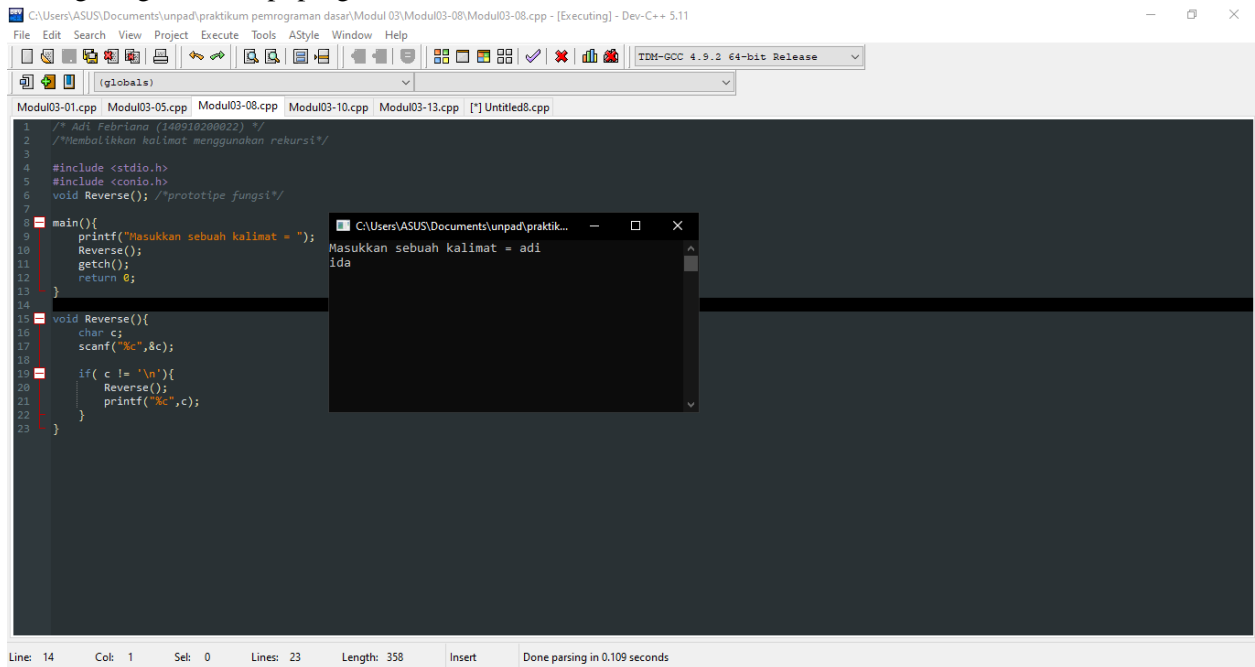
```
1  /* Adi Febriana (140910200022) */  
2  /*Menghitung jumlah dari bilangan asli dengan menggunakan rekursi*/  
3  
4  #include <stdio.h>  
5  #include <conio.h>  
6  int tambah(int n); /*prototipe fungsi*/  
7  
8  main()  
9  {  
10     int n;  
11     printf("Masukkan sebuah integer positif = ");  
12     scanf("%d",&n);  
13     printf("Jumlah = %d",tambah(n));  
14  
15     getch();  
16     return 0;  
17 }  
18  
19 int tambah(int n){  
20     if(n!=0)  
21         return n+tambah(n-1); /* pemanggilan rekursif */  
22 }
```

Line: 10 Col: 5 Sel: 0 Lines: 22 Length: 427 Insert Done parsing in 0.032 seconds

Analisis :

Pada baris 6, prototipe fungsi tambah di deklarasi. Pada baris 9, dideklarasikan variabel n dengan tipe data integer. Pada baris 12, data integer positif diinput ke variabel n. Pada baris 13, hasil dari fungsi tambah ditampilkan di konsol. Pada fungsi tambah, pada baris 20 terjadi pengecekan dengan kondisi n tidak sama dengan 0. Pada kasus diatas, nilai integer yang diinput adalah 5, lalu pada baris 13 dipanggil fungsi tambah dengan parameter 5. Pada baris 20, kondisi n tidaksama dengan 0, dan kondisi itu terpenuhi dan pada baris 21 nilai n + tambah(n-1) (pemanggilan fungsi tambah dengan parameter n-1), nilai dikembalikan, lalu masuk lagi ke fungsi tambah dengan parameter (n-1), hingga ketika parameter $n-1 = 0$ tidak terjadi pengembalian nilai karena kondisi sudah tidak terpenuhi. dan nilai yang akan tampil ialah, 15, $(5 + (4 + (3 + (2 + (1+0))))))$.

1.3 Listing Program / Script program



The screenshot shows a C++ IDE with a file named `Modul03-08.cpp` open. The code implements a recursive function `Reverse` to reverse a string. The `main` function prompts the user to enter a string, which is then reversed and printed. The output window shows the input "Masukkan sebuah kalimat = adi" and the output "ida".

```
1  /* Adi Febriana (140910200022) */  
2  /* Membalikkan kalimat menggunakan rekursi */  
3  
4  #include <stdio.h>  
5  #include <conio.h>  
6  void Reverse(); /*prototipe fungsi*/  
7  
8  int main(){  
9      printf("Masukkan sebuah kalimat = ");  
10     Reverse();  
11     getch();  
12     return 0;  
13 }  
14  
15 void Reverse(){  
16     char c;  
17     scanf("%c",&c);  
18  
19     if( c != '\n'){  
20         Reverse();  
21         printf("%c",c);  
22     }  
23 }
```

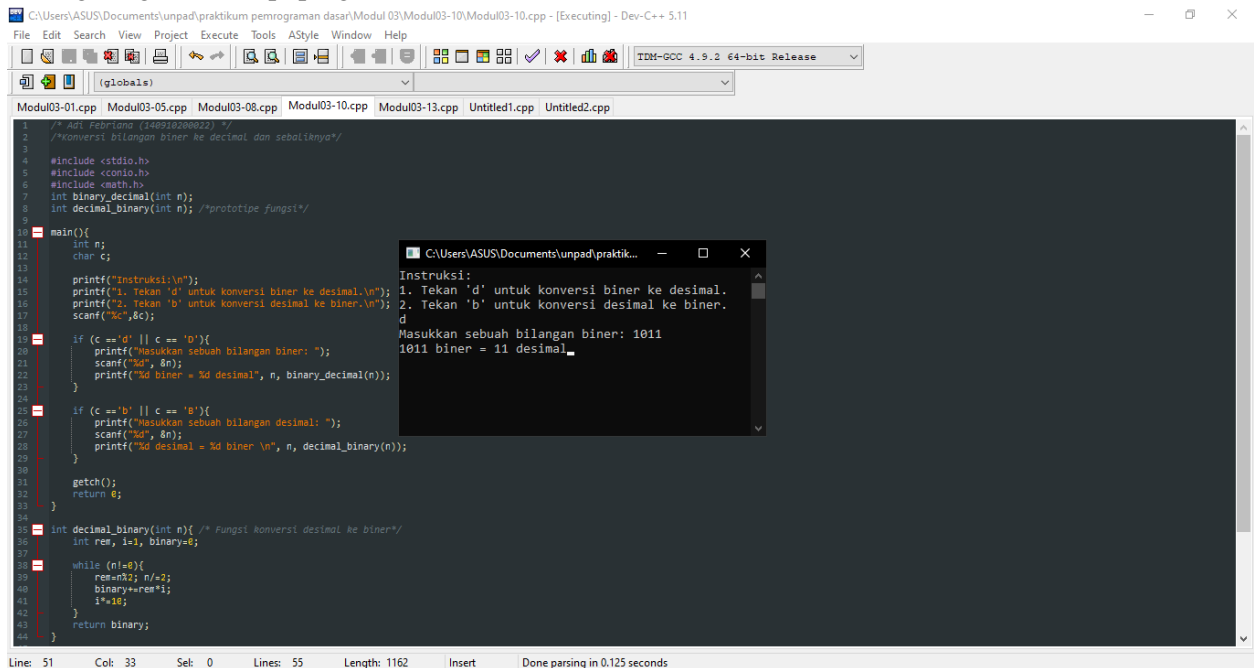
Output window: C:\Users\ASUS\Documents\unpad\praktik...
Masukkan sebuah kalimat = adi
ida

Line: 14 Col: 1 Sel: 0 Lines: 23 Length: 358 Insert Done parsing in 0.109 seconds

Analisis :

Pada baris 6, prototipe reverse dideklarasikan. Tidak ada pendeklarasian variabel di blok kode main. Pada baris 10, fungsi reverse dipanggil. Dalam fungsi reverse tersebut, pada baris 16, variabel `c` dideklarasikan dengan tipe data karakter. Lalu pada baris 17 data kalimat dari pengguna diinputkan ke variabel `i`. pada baris 19 terjadi pengecekan kondisi, ketika variabel `c` tidak sama dengan “`\n`“, pada kasus diatas, kalimat yang diinput ialah “Adi”, karena variabel `c` bertipe data `char`, maka yang akan tersimpan terlebih dahulu ialah huruf a. lalu pada baris 20 terjadi pemanggilan fungsi reverse lagi, karena huruf a sudah tersimpan dan sisa hurufnya tinggal d dan I, maka selanjutnya yang tersimpan ialah huruf d lalu terjadi pengulangan lagi dan yang tersimpan huruf i. Setelah semua huruf sudah tersimpan, tidak terjadi lagi pemanggilan fungsi reverse karena kondisi pada baris 19 sudah tidak terpenuhi. lalu program akan keluar dari rekursi terdalam dan menampilkan di konsol, dengan urutan yakni huruf I, dilanjut dengan d, dilanjut dengan a. Sehingga yang tampil dikonsol adalah ida.

1.4 Listing Program / Script program



```
1  /* Fungsi konversi biner ke desimal */
2  /* Konversi bilangan biner ke desimal dan sebaliknya */
3
4  #include <stdio.h>
5  #include <conio.h>
6  #include <math.h>
7  int binary_decimal(int n);
8  int decimal_binary(int n); /*prototipe fungsi*/
9
10 main(){
11     int n;
12     char c;
13
14     printf("Instruksi:\n");
15     printf("1. Tekan 'd' untuk konversi biner ke desimal.\n");
16     printf("2. Tekan 'b' untuk konversi desimal ke biner.\n");
17     scanf("%c",&c);
18
19     if (c == 'd' || c == 'D'){
20         printf("Masukkan sebuah bilangan biner: ");
21         scanf("%d",&n);
22         printf("Biner = %d desimal", n, binary_decimal(n));
23     }
24
25     if (c == 'b' || c == 'B'){
26         printf("Masukkan sebuah bilangan desimal: ");
27         scanf("%d",&n);
28         printf("Desimal = %d biner \n", n, decimal_binary(n));
29     }
30
31     getch();
32     return 0;
33 }
34
35 int decimal_binary(int n){ /* Fungsi konversi desimal ke biner */
36     int rem, i=1, binary=0;
37
38     while (n!=0){
39         rem=n%2; n/=2;
40         binary+=rem*i;
41         i*=10;
42     }
43     return binary;
44 }
```

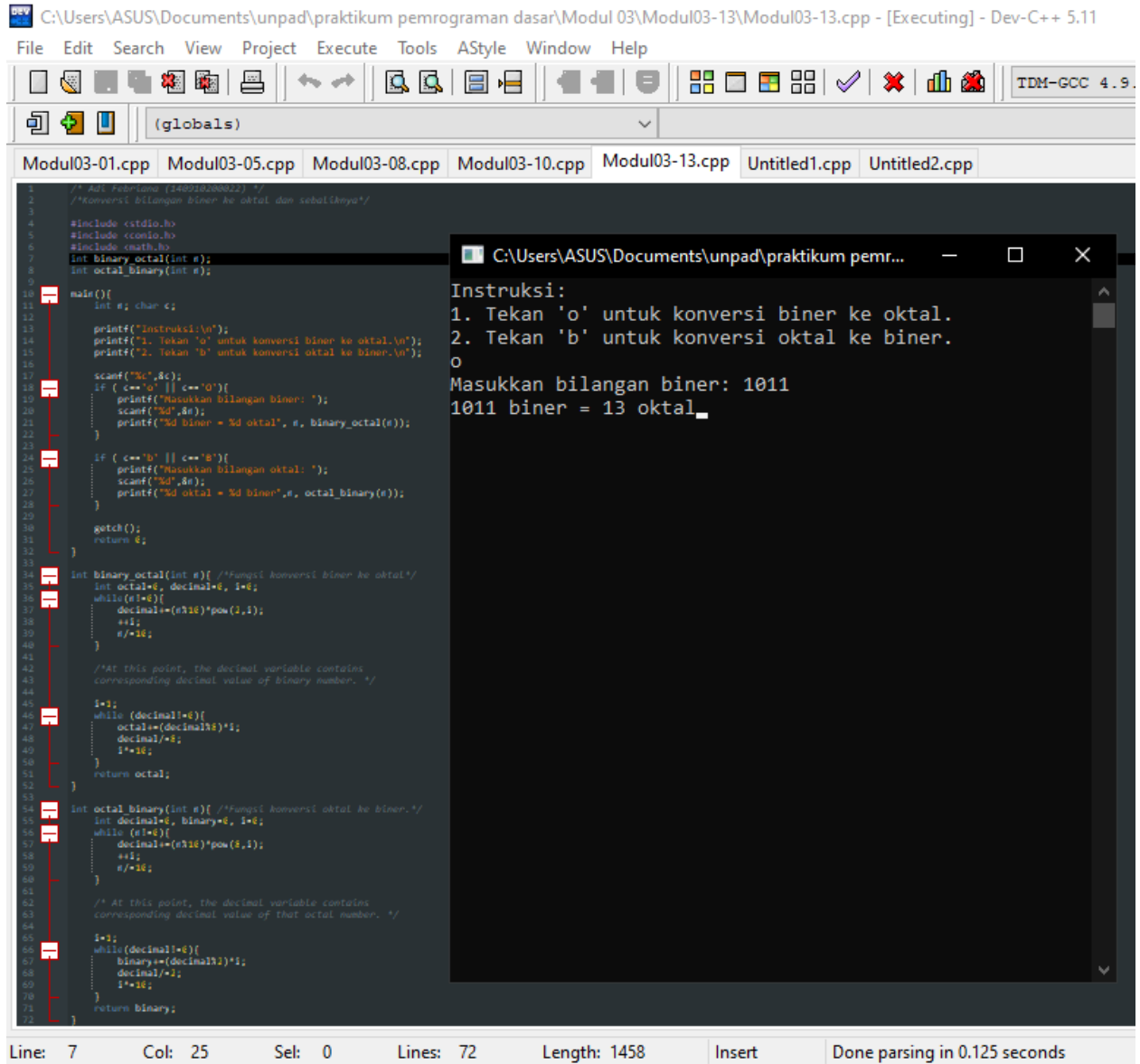
Analisis :

Pada baris 7 dan 8, fungsi `binary_decimal(int n)` dan fungsi `decimal_binary(int n)` dideklarasikan. Lalu pada baris 11 dan 12 dideklarasikan variabel `n` yang bertipe data integer dan variabel `c` yang bertipe data karakter. Pada baris 17, data huruf diinputkan kedalam variabel `c`. ketika pengguna memasukkan huruf `d`, maka akan masuk ke blok kode `if` (baris 19 – 23), dan memanggil fungsi `binary_decimal`. Sedangkan ketika pengguna memasukkan huruf `b`, maka akan masuk ke blok kode `if` (baris 25-29), dan memanggil fungsi `decimal_binary`. Pada kasus diatas, pengguna memasukkan huruf `d`, lalu akan kode blok `if`(baris 19-23) dieksekusi. Lalu pengguna memasukkan nilai biner 1011, dan pada baris 22, fungsi `binary_decimal` dengan parameter 1011, dipanggil. Pada blok kode fungsi `binary_decimal` (46-55), di deklarasi variabel `decimal` dengan nilai 0, variabel `i` dengan nilai 0, dan variabel `rem` dengan tipe data integer. Lalu pada baris 48, terjadi pengulangan, dan pengulangan itu terpenuhi karena `n = 1011`. Pengulangan terjadi, `rem` akan bernilai 1, `n` bernilai 101, dan `decimal` bernilai 1, dan variabel `i` akan bertambah. Pengulangan akan terus terjadi hingga nilai `n = 0`, ketika itu kondisi sudah tidak terpenuhi dan program akan keluar dan yang tampil di konsol adalah 11.

Jika pada baris 17, pengguna memasukkan huruf `b`, lalu akan kode blok `if`(baris 25-29) dieksekusi. Lalu pengguna memasukkan nilai desimal 11, dan pada baris 28, fungsi `decimal_binary` dengan parameter 11, dipanggil. Pada blok kode fungsi `decimal_binary` (35-44), di deklarasi variabel `binary` dengan nilai 0, variabel `i` dengan nilai 1, dan variabel `rem` dengan tipe data integer. Lalu pada baris 38, terjadi pengulangan, dan pengulangan itu terpenuhi karena `n = 11`. Pengulangan terjadi, `rem` akan bernilai 1, `n` bernilai 5, dan `binary` bernilai 1, dan variabel `i` akan dikali 10. Pengulangan akan terus

terjadi hingga nilai $n = 0$, ketika itu kondisi sudah tidak terpenuhi dan program akan keluar dan yang tampil di konsol adalah 1011.

1.5 Listing Program / Script program



```
1  /* Adi Febriana (14011400012) */
2  /*Konversi bilangan biner ke oktal dan sebaliknya*/
3
4  #include <stdio.h>
5  #include <conio.h>
6  #include <math.h>
7  int binary_octal(int n);
8  int octal_binary(int n);
9
10 main(){
11     int n; char c;
12
13     printf("Instruksi:\n");
14     printf("1. Tekan 'o' untuk konversi biner ke oktal.\n");
15     printf("2. Tekan 'b' untuk konversi oktal ke biner.\n");
16
17     scanf("%c",&c);
18     if ( c=='o' || c=='O'){
19         printf("Masukkan bilangan biner: ");
20         scanf("%d",&n);
21         printf("Bilangan biner = %d oktal", n, binary_octal(n));
22     }
23     if ( c=='b' || c=='B'){
24         printf("Masukkan bilangan oktal: ");
25         scanf("%d",&n);
26         printf("Bilangan oktal = %d biner", n, octal_binary(n));
27     }
28     getch();
29     return 0;
30 }
31
32
33 int binary_octal(int n){ /*Fungsi konversi biner ke oktal*/
34     int decimal=0, binary=n, i=0;
35     while(n!=0){
36         decimal+=(n%10)*pow(2,i);
37         n/=10;
38         i++;
39     }
40     /*At this point, the decimal variable contains
41     corresponding decimal value of binary number. */
42     int oktal=0;
43     while (decimal!=0){
44         oktal+=(decimal%8)*i;
45         decimal/=8;
46         i++;
47     }
48     return oktal;
49 }
50
51 int octal_binary(int n){ /*Fungsi konversi oktal ke biner*/
52     int decimal=0, binary=n, i=0;
53     while(n!=0){
54         decimal+=(n%8)*pow(2,i);
55         n/=8;
56         i++;
57     }
58     /* At this point, the decimal variable contains
59     corresponding decimal value of that octal number. */
60     int biner=0;
61     while (decimal!=0){
62         biner+=(decimal%2)*i;
63         decimal/=2;
64         i++;
65     }
66     return biner;
67 }
68
69
70
71
72
```

Line: 7 Col: 25 Sel: 0 Lines: 72 Length: 1458 Insert Done parsing in 0.125 seconds

C:\Users\ASUS\Documents\unpad\praktikum pemr... — □ ×

Instruksi:
1. Tekan 'o' untuk konversi biner ke oktal.
2. Tekan 'b' untuk konversi oktal ke biner.
o
Masukkan bilangan biner: 1011
1011 biner = 13 oktal_

Analisis :

Pada baris 7 dan 8, fungsi `binary_octal(int n)` dan fungsi `octal_binary(int n)` dideklarasikan. Lalu pada baris 11 dideklarasikan variabel `n` yang bertipe data integer dan variabel `c` yang bertipe data karakter. Pada baris 17, data huruf diinputkan kedalam variabel `c`. ketika pengguna memasukkan huruf `o`, maka akan masuk ke blok kode `if` (baris 18 – 22), dan memanggil fungsi `binary_octal`. Sedangkan ketika pengguna memasukkan huruf `b`, maka akan masuk ke blok kode `if` (baris 24-28), dan memanggil fungsi

octal_binary. Pada kasus diatas, pengguna memasukkan huruf o, lalu akan kode blok if(baris 18-22) dieksekusi. Lalu pengguna memasukkan nilai biner 1011, dan pada baris 21, fungsi binary_octal dengan parameter 1011, dipanggil. Pada fungsi binary_octal, dideklarasikan variabel ocktal dengan nilai 0, decimal dengan nilai 0, dan variabel I dengan nilai 0. Lalu pada baris 36 terjadi pengulangan dan kondisi terpenuhi karena n=1011. Pengulangan terjadi, decimal akan berubah nilai 1, lalu I akan bertambah menjadi 1 dan nilai n bernilai 1, dan ini akan berulang hingga n bernilai 0. Dan nilai decimal setelah keluar dari perulangan bernilai 11. Pada baris 45, nilai I diubah menjadi 1 dan pada baris 46, perulangan terjadi. Perulangan terjadi, octal akan berubah menjadi bernilai 3, decimal menjadi 1 dan I akan menjadi 10. Perulangan akan terus terjadi hingga decimal = 0. Dan keluar dari perulangan tersebut dan nilai octal dikembalikan dan pada konsol akan tampil 1011 biner = 13 oktal. Pada program ini, biner diubah menjadi desimal dan desimal akan menjadi oktal.

Jika pengguna memasukkan huruf b pada baris 17, kode blok if(baris 24-28) dieksekusi. Lalu pengguna memasukkan nilai biner 13, dan pada baris 27, fungsi octal_binary dengan parameter 13, dipanggil. Pada fungsi octal_binary, dideklarasikan variabel decimal dengan nilai 0, binary dengan nilai 0, dan variabel I dengan nilai 0. Lalu pada baris 56 terjadi pengulangan dan kondisi terpenuhi karena n=13. Pengulangan terjadi, decimal akan berubah nilai 31, lalu I akan bertambah menjadi 1 dan nilai n bernilai 1, dan ini akan berulang hingga n bernilai 0. Dan nilai decimal setelah keluar dari perulangan bernilai 11. Pada baris 65, nilai I diubah menjadi 1 dan pada baris 66, perulangan terjadi. Perulangan terjadi, binary akan berubah menjadi bernilai 1, decimal menjadi 5 dan I akan menjadi 10. Perulangan akan terus terjadi hingga decimal = 0, dan keluar dari perulangan tersebut dan nilai binary dikembalikan dan pada konsol akan tampil 13 oktal = 1011 biner. Pada program ini, oktal diubah menjadi desimal dan desimal akan menjadi biner.

Kesimpulan

Jadi, setelah melakukan praktikum modul 3, 3.3.1, 3.3.5, 3.3.8, 3.3.10, dan 3.3.13, praktikan mampu memecahkan program besar yang kompleks menjadi program-program modular yang lebih kecil dengan benar.