

Muhammad Niko

```
1 keluar := "N";
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```



Buku Saku Praktikum Bahasa Pemrograman PASCAL



Buku Saku Praktikum : Bahasa Pemrograman Pascal

MUHAMMAD NIKO

Buku ini dipublikasikan di bawah Lisensi
[Creative Commons Attribution-ShareAlike 4.0 International](https://creativecommons.org/licenses/by-sa/4.0/)
(CC BY-SA 4.0)

Agustus 2018
55 Halaman, A4

Kata Pengantar

Bahasa pemrograman pascal sangat cocok dipelajari sebagai pengantar sebelum mempelajari bahasa pemrograman tingkat lanjut. Jika anda telah memahami pemrograman pascal, dan kemudian memutuskan untuk mempelajari bahasa pemrograman lain, diharapkan anda tidak lagi kesulitan, karena sudah terbiasa menyusun “kode program” serta algoritma ketika mempelajari pemrograman pascal.

Dalam mempelajari bahasa pemrograman pascal, Kegiatan praktikum sangat penting untuk dilakukan. dalam buku ini materi praktikum bahasa pemrograman pascal telah disusun secara urut, dimulai dari tingkat yang paling dasar.

Dengan membaca buku ini diharapkan para pembaca dapat belajar memahami serta melakukan praktikum bahasa pemrograman pascal secara mandiri maupun dibantu oleh pengajar.

Banjarmasin, Agustus 2018

Muhammad Niko

Tentang Penulis

Muhammad Niko adalah seorang Mahasiswa program studi S1 Sistem Informasi dan juga Alumni SMK Negeri jurusan Teknik Komputer dan Jaringan. Memiliki minat yang tinggi terhadap IT khususnya pada bidang Web Desain serta Bahasa Pemrograman.

Terkadang juga aktif menulis artikel di blog pribadi miliknya yang dapat diakses melalui alamat www.muhammadniko.web.id. Saat ini telah menulis serta mempublikasikan beberapa buku yang dapat didownload secara gratis.

Saran, Pertanyaan, dan Kerja Sama :

LinkedIn : <https://www.linkedin.com/in/muhammadniko98/>

Email : mn.muhammadniko@gmail.com

Situs Web : <https://www.muhammadniko.web.id/>

Daftar Isi

Kata Pengantar.....	3
Tentang Penulis.....	4
Materi 1 : Berkenalan dengan Bahasa Pemrograman Pascal.....	6
1.1 Mengenal Pemrograman Pascal.....	6
1.2 Belajar Pemrograman Pascal, Untuk Apa ?.....	6
1.3 Struktur Penulisan Pemrograman Pascal.....	7
Materi 2 : Memulai Praktikum Menggunakan Free Pascal.....	9
2.1 Cara Install Aplikasi Free Pascal.....	9
2.2 Memulai membuat Program dengan Free Pascal.....	12
Materi 3 : Praktikum Bahasa Pemrograman Pascal.....	14
3.1 Output (Menampilkan) Data.....	14
3.2 Variabel dan Tipe Data.....	15
3.3 Input (Memasukan) Data.....	18
3.4 Baris Komentar	19
3.5 Operasi Perhitungan Arimatika.....	20
3.6 Operasi Perbandingan.....	21
3.7 Operasi Logika.....	22
3.8 Struktur Penyeleksian.....	23
3.9 Struktur Perulangan.....	29
3.10 Array.....	36
3.11 Array 2 Dimensi.....	39
3.12 Modular Programming.....	42
Materi 4 : Studi Kasus – Membuat Program Kasir.....	48

Materi 1 Berkenalan Dengan Bahasa Pemrograman Pascal

1.1 Mengenal Pemrograman Pascal

Bahasa Pemrograman Pascal pertama kali dikembangkan pada tahun 1971 oleh Professor Niklaus Wirth. Kata pascal diambil dari nama seorang ilmuwan matematika bernama Blaise Pascal. Bahasa Pascal termasuk dalam Bahasa Pemrograman Tingkat Tinggi, sehingga perintah-perintah yang digunakan dalam bahasa pemrograman ini sangat terstruktur serta sistematis.

Untuk membuat program menggunakan bahasa pascal, diperlukan aplikasi Compiler yang dapat mengkompilasi bahasa pascal menjadi sebuah program utuh. Compiler Pascal yang sudah sangat populer untuk digunakan adalah Turbo Pascal yang dikembangkan oleh Borland. Sayangnya, Turbo Pascal merupakan aplikasi berbayar serta pengembangannya telah dihentikan sejak bertahun-tahun yang lalu.

Selain Turbo Pascal, masih banyak Aplikasi compiler pascal yang dapat anda gunakan, salah satunya adalah Free Pascal, yang merupakan compiler pascal *open source* yang masih dikembangkan hingga saat ini.

1.2 Belajar Pemrograman Pascal, Untuk Apa ?

Melihat perkembangan berbagai macam bahasa pemrograman yang ada, mungkin pemrograman pascal sudah sangat jauh tertinggal. Bahkan untuk saat ini dalam membangun sebuah sistem ataupun aplikasi, bahasa pemrograman pascal sudah sangat jarang digunakan. Lalu, untuk apa dipelajari ?

Dalam dunia pendidikan, Pemrograman pascal dipelajari untuk mempermudah seseorang dalam mengenal "Programming". Biasanya, sebelum diajarkan bahasa pemrograman tingkat lanjut, para pelajar di tingkat SMK bahkan Perguruan Tinggi, mereka akan terlebih dahulu diajarkan dasar-dasar Pemrograman dan Algoritma yang dipraktik-kan dalam bahasa pascal.

Hal itu dikarenakan bahasa pemrograman pascal termasuk bahasa pemrograman yang cukup mudah untuk dipelajari serta notasi penulisannya tidak begitu rumit.

Jika anda mengamati setiap perintah pada bahasa pemrograman pascal, ternyata tidak jauh berbeda dengan kata-kata yang ada dalam bahasa inggris. Dengan begitu seharusnya anda dapat dengan mudah menterjemahkan bahasa komputer (pascal) kedalam bahasa manusia ataupun sebaliknya.

Misalnya pada Contoh penggalan program bahasa pascal dibawah ini :

```
Begin
Umur_saya := 21
if umur_saya > 18 then
  write('saya Telah Dewasa')
end.
```

Dapat anda baca kedalam bahasa manusia, menjadi :

```
mulai
Umur saya adalah 21
Jika umur saya lebih dari 18, maka
  Tulis "saya telah dewasa"
selesai
```

Tidak sulit untuk dipahami bukan ? pastinya anda sudah dapat membayangkan bagaimana pelajaran yang akan anda hadapi kedepannya, semangat ! dan Selamat melanjutkan bacaan anda :)

1.3 Struktur Penulisan Bahasa Pascal

Secara umum program pascal memiliki 2 bagian struktur, yaitu header dan body program. Header adalah bagian untuk meletakkan judul program, pemanggilan unit, serta pendeklarasian (variabel, procedure, function, dll). Sedangkan bagian body untuk menuliskan perintah-perintah yang akan menjadi program utama saat dijalankan.

setiap anda menulis program dalam bahasa pascal, maka struktur penulisan yang akan selalu anda gunakan adalah sebagai berikut :

```
program nama_program;
uses crt;
begin
end.
```

1. Program nama_program;

digunakan untuk menginformasikan nama atau judul dari program yang anda tulis kepada compiler, perintah ini selalu diletakan dibaris paling awal. Dalam penulisannya tidak boleh mengandung karakter unik dan untuk setiap kata hanya dapat dipisahkan dengan simbol garis bawah “_”.

2. Uses crt;

Digunakan untuk memanggil unit crt yang berisi kumpulan fungsi standar program. Ada banyak unit yang dapat anda gunakan sesuai kebutuhan. Unit yang paling umum digunakan dalam membuat program pascal sederhana adalah WINCRT atau CRT.

3. Begin

Merupakan perintah yang menunjukkan awal dimana program akan mulai dijalankan. instruksi yang akan dikerjakan oleh program utama, nantinya akan ditulis diantara perintah “Begin” dan “end.”

4. End.

Merupakan perintah untuk menghentikan seluruh eksekusi dan akan mengakhiri seluruh program yang anda tulis. Perintah ini selalu ditulis dibaris paling bawah program anda.

Materi 2 Memulai Praktikum Menggunakan Free Pascal

Sebelumnya anda telah mengetahui bahwa, untuk dapat menjalankan bahasa pascal dibutuhkan sebuah aplikasi *pascal compailer*. Pada buku ini, kita akan melakukan praktikum menggunakan Free Pascal sebagai aplikasi compailernya.

Berikut cara melakukan instalasi Free Pascal :

2.1. Cara Install Aplikasi Free Pascal

Download Installer Free Pascal

Silahkan anda download terlebih dahulu installer untuk program free pascal melalui situs resminya di : <https://freepascal.org/download.html/>

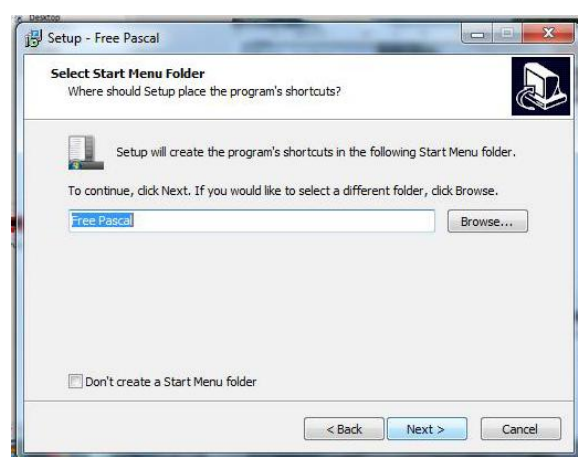
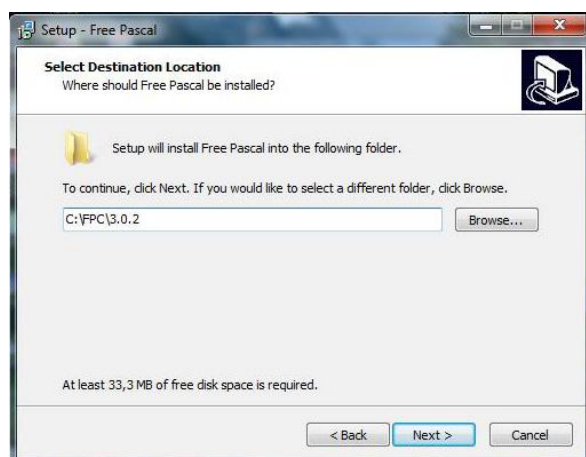
Pilih versi yang sesuai dengan sistem operasi yang anda gunakan.

Memulai Instalasi Free Pascal

Buka file installer Free Pascal yang telah anda download sebelumnya. Maka Akan muncul tampilan seperti gambar dibawah ini. Klik "Next".



Pada Bagian ini anda dapat memilih direktori dimana program Free Pascal akan diinstall. Secara default Free Pascal akan ter-install pada direktori C:/FPC/. Klik "Next" untuk melanjutkan

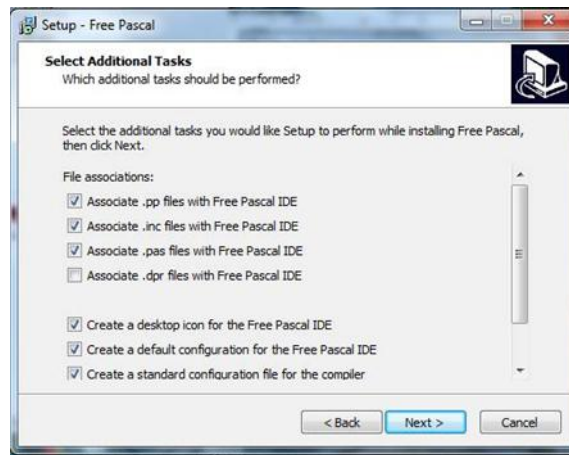


Selanjutnya silahkan anda Pilih mode Instalasi. pilih "Full Installation", Agar semua

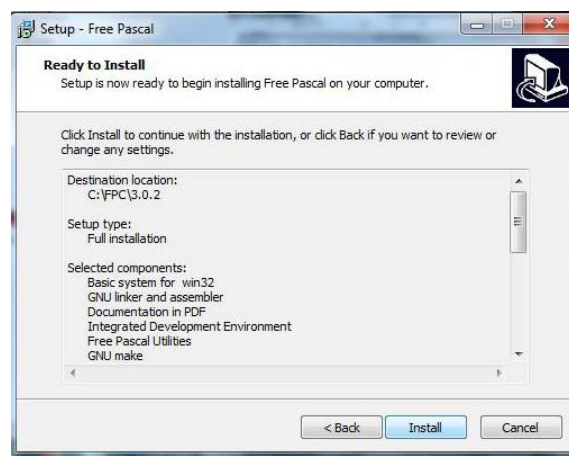


kebutuhan program dapat terinstall semua, kemudian klik “Next”.

Berikan tanda centang pada bagian “Associate .pas files with Free Pascal IDE” sehingga file dengan format .pas dapat terintegrasi dengan Free Pascal. Lalu klik “Next”.



Klik “Install” untuk memulai instalasi. Tunggu beberapa saat hingga aplikasi Free Pascal selesai melakukan instalasi. Setelah itu klik “Finish”.



Sampai disini Aplikasi Free Pascal telah berhasil terinstall di komputer anda. Untuk menjalankannya, dapat melalui shortcut yang ada pada Desktop.

2.2 Memulai Membuat Program dengan Free Pascal

Membuat File Project Baru

Buka program Free Pascal yang telah terinstall di komputer anda, lalu buat file project baru dengan cara klik menu “File” → “New”.

anda dapat menuliskan kode program pada tampilan text editor yang tersedia. Untuk mencobanya silahkan anda tuliskan terlebih dahulu kode program dibawah ini :

```
program pascal_pertama;
uses crt;
begin
  clrscr;
  write('Hello World !!');
  readln;
end.
```

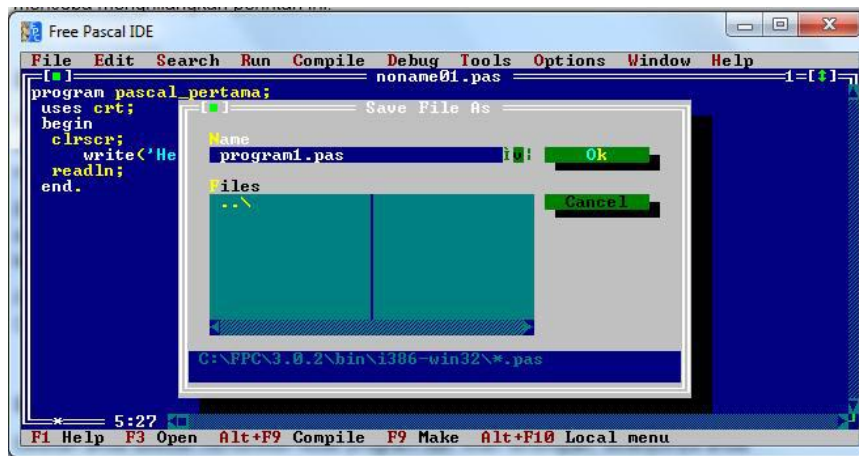
Penjelasan : kode program diatas akan menampilkan tulisan “Hello World !!” pada layar. Jika anda lihat, pada bagian program utama (body) terdapat 3 buah perintah yang akan dijalankan yaitu :

1. Clrscr – perintah ini berfungsi untuk membersihkan layar dari tampilan program yang telah dijalankan sebelumnya. Anda dapat mencoba menghilangkan perintah ini untuk melihat pengaruhnya pada program.
2. Write – perintah untuk menampilkan output ke layar. Perintah ini akan dijelaskan lebih mendalam pada pembahasan selanjutnya.
3. Readln – Dalam hal ini perintah readln digunakan untuk mencegah layar program anda menutup saat dijalankan. Untuk melihat pengaruhnya pada program, anda dapat mencoba menghilangkan perintah ini.

Menyimpan Kode Program (SAVE)

Kode program yang ditulis dalam bahasa pascal akan disimpan dalam format “.pas”. Untuk menyimpan kode program yang telah anda buat dapat dilakukan dengan cara klik menu “File” → “Save”.

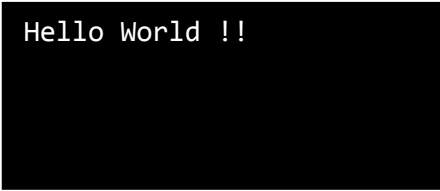
Beri nama file pascal anda, jangan lupa menambahkan format “.pas” di akhir nama file agar dapat dikenali sebagai kode program pascal. kemudian klik “OK”.



File project pascal anda secara default akan disimpan pada direktori
C:\FPC\3.0.2\bin\i386-win32

Menjalankan Program Pascal (RUN)

Setelah anda selesai menuliskan kode program dan telah tersimpan, selanjutnya anda dapat mencoba untuk menjalankan program yang anda buat dengan cara klik menu “Run” → “Run”, atau anda dapat menekan tombol “CTRL + F9” pada keyboard. Tunggu beberapa saat hingga program anda tampil.



Hello World !!

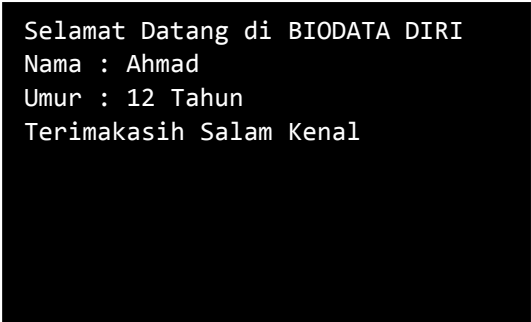
Materi 3 Praktikum Bahasa Pemrograman Pascal

3.1 Output (Menampilkan) Data

Untuk menampilkan output berupa tulisan pada layar, dapat menggunakan perintah "WRITE" dan "WRITELN". Perbedaan antara kedua perintah tersebut ada pada tampilan yang dihasilkan. Jika menggunakan perintah "write" maka tulisan yang dihasilkan akan berada dalam satu baris yang sama (inline), sedangkan jika menggunakan perintah "writeln" maka tulisan yang ada setelahnya akan berganti baris dan berada pada baris berikutnya (new line).

Contoh Program :

```
program biodata;
uses crt;
begin
  clrscr;
  write('Selamat Datang di ');
  write('BIODATA DIRI');
  writeln;
  writeln('Nama : Ahmad');
  writeln('Umur : 12 Tahun');
  writeln;
  write('Terimakasih ');
  write('Salam Kenal');
  readln;
end.
```



```
Selamat Datang di BIODATA DIRI
Nama : Ahmad
Umur : 12 Tahun
Terimakasih Salam Kenal
```

Penjelasan : perhatikan hasil dari program diatas. Tulisan yang ditampilkan menggunakan perintah write akan berada dalam satu baris yang sama. Sedangkan yang ditampilkan menggunakan perintah writeln berada pada baris baru dibawahnya.

3.2 Variabel dan Tipe Data pada Pascal

Variabel digunakan untuk menyimpan suatu data secara sementara didalam memori komputer. Setiap variabel memiliki tipe data tertentu sesuai dengan data yang akan disimpan dalam variabel tersebut.

Dalam pemrograman pascal terdapat banyak tipe data yang dapat anda gunakan sesuai kebutuhan program yang anda buat. Adapun dalam buku ini hanya menuliskan beberapa tipe data yang umumnya digunakan dalam membuat program pascal sederhana, seperti yang dapat anda lihat pada tabel dibawah ini.

Tipe Data	Jangkauan Nilai	Jenis	Contoh Nilai
Byte	0 s/d 255	Numerik, Bilangan Bulat	5
Integer	-32768 s/d 32767	Numerik, Bilangan Bulat	15500
Longint	-2147483648 s/d 2147483647	Numerik, Bilangan Bulat	12500000
Real	3.4×10^{-38} s/d 3.4×10^{38}	Numerik, Bilangan Desimal	3.14
Char	Seluruh Karakter ASCII	Sebuah Karakter	*
String	Seluruh Karakter ASCII	Kumpulan Karakter	Salam Kenal

Deklarasi Variabel

Sebelum dapat digunakan, sebuah variabel harus terlebih dahulu “dideklarasikan” agar dapat dikenali oleh komputer. Pada pemrograman pascal, variabel dideklarasikan dengan perintah “VAR” yang diletakkan pada bagian header program sebelum perintah “Begin”.

Bentuk umum penulisan :

```
var nama_variabel : tipe_data;
```

Contoh : `var nama_siswa : string;`

Variabel dengan tipe data yang sama dapat ditulis dalam satu baris, Setiap item variabel dipisahkan dengan tanda koma “,”

```
var variabel_1, variabel_2, variabel_3 : tipe_data;
```

Contoh : `var umur, nomor_hp : integer;`

Variabel dengan tipe data yang berbeda-beda dapat dituliskan pada baris berikutnya

```
var
    variabel_1 : tipe_data_1;
    variabel_2 : tipe_data_2;
```

Contoh :

```
var
    harga, diskon : integer;
    nomor : byte;
    pesan : string
```

Dalam penamaan variabel terdapat hal-hal yang perlu diperhatikan seperti, nama variabel hanya dapat diawali dengan huruf atau garis bawah “_”, tidak boleh mengandung spasi kosong, dan nama variabel hanya boleh terdiri dari huruf angka dan garis bawah.

Memberikan Nilai Pada Variabel

Untuk memberikan nilai pada suatu variabel dalam pemrograman pascal, digunakan simbol titik dua sama dengan “:=” (Assignment Operator), Nilai yang akan disimpan pada variabel harus sesuai dengan tipe data dari variabel tersebut.

Bentuk umum penulisan : `nama_variabel := isi_variabel;`

Contoh penulisan :


`x := 9;` variabel x akan berisi nilai 9.

`pesan := 'Belajar yang rajin';`

variabel pesan akan berisi nilai string "Belajar yang rajin".

Contoh Program :

```
program contoh_variabel;
uses crt;
var
  no_induk      : integer;
  nama_karyawan : string;
begin
  clrscr;
  no_induk:=8704;
  nama_karyawan := 'Muhammad Effendi';
  writeln(nama_karyawan);
  writeln(no_induk);
  readln;
end.
```



Muhammad Effendi
8704

Penjelasan : pada contoh diatas kita mendeklarasikan 2 buah variable (no_induk dan nama_karyawan), kemudian pada program utama kita berikan nilai pada masing-masing variabel, terakhir kita tampilkan kedua isi variabel tadi menggunakan perintah writeln.

Tip : Perlu anda ingat, dalam bahasa pemrograman pascal setiap data yang berupa karakter / teks, maka harus diletakkan di antara dua buah tanda petik atas.

Contoh :

```
Writeln ('Belajar Pemrograman Pascal');
Pesan := 'Hati-hati dijalan';
```

3.3 Input (Memasukkan) Data

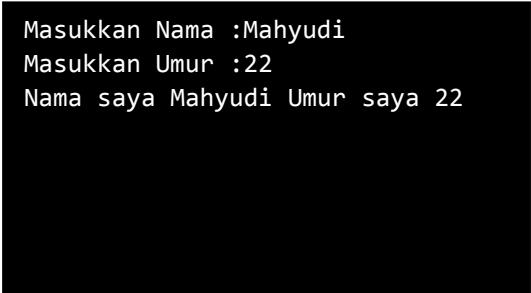
Proses Input (Memasukkan) data pada pemrograman pascal dapat dilakukan menggunakan perintah “readln”. Setiap data yang diinputkan melalui perintah readln nantinya akan disimpan dalam sebuah variabel. Oleh karena itu perintah readln selalu diikuti dengan variabel untuk menampung data yang telah diinput-kan.

Bentuk umum penulisan : `readln(nama_variabel);`

Contoh Program :

```
program contoh_program_input;
uses crt;
var
  nama : string;
  umur : integer;
```

```
Begin
  clrscr;
  write('Masukkan Nama :'); readln(nama);
  write('Masukkan Umur :'); readln(umur);
  writeln ('Nama saya ', nama, ' Umur saya ', umur);
readln;
end.
```



Masukkan Nama :Mahyudi
Masukkan Umur :22
Nama saya Mahyudi Umur saya 22

Penjelasan: pada contoh program diatas terdapat 2 buah proses input-an, nantinya data yang telah dimasukkan akan disimpan dalam variabel “nama” dan “umur”, kemudian isi dari variabel “nama” dan “umur” akan ditampilkan menggunakan perintah writeln.

Tip : anda dapat menampilkan beberapa data (dengan tipe data yang berbeda) dalam satu baris perintah writeln. setiap data yang ingin ditampilkan harus dipisahkan dengan tanda koma. Seperti yang ada pada baris `writeln ('Nama saya ', nama, ' Umur saya ', umur).`

3.4 Baris Komentar

Baris komentar digunakan untuk memberikan keterangan pada kode program yang anda tulis. Semua yang anda tuliskan didalam Baris komentar tidak akan dijalankan oleh program. Baris komentar di tulis menggunakan simbol double slash “//”

Contoh penulisan : `// ini keterangan kode`

Baris komentar juga dapat berbentuk blok komentar yang ditulis diantara simbol “{” dan “}”, sehingga dapat memuat lebih dari satu baris keterangan.

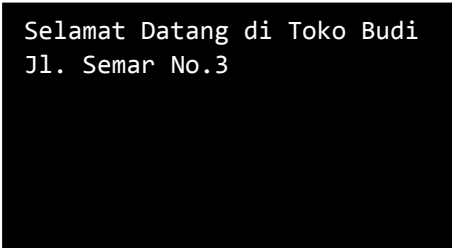
Contoh penulisan :

```
{  
ini adalah blok komentar  
hanya sebagai keterangan  
}
```

Pada aplikasi free pascal, baris komentar akan ditandai dengan warna abu-abu.

Contoh Program

```
program baris_komentar;  
uses crt;  
  
begin  
clrscr;  
writeln ('Selamat Datang di Toko Budi');  
writeln ('Jl. Semar No.3'); //tampilkan nama jalan  
  
{  
writeln ('Hp. 0821001010');  
writeln ('Website : www.budi.com');  
}  
  
readln;  
end.
```



```
Selamat Datang di Toko Budi  
Jl. Semar No.3
```

Penjelasan : pada contoh program diatas, no hp dan website tidak ditampilkan karena berada dalam blok komentar.

3.5 Operasi Perhitungan Arimatika

Untuk melakukan perhitungan aritmatika dalam pemrograman pascal, diperlukan beberapa Operator Artikmatika seperti yang ada pada tabel dibawah ini :


Operator	Fungsi	Contoh
+	Penjumlahan	$x := 1 + 2;$
-	Pengurangan	$x := 4 - 2;$
*	Perkalian	$x := 3 * 5;$
/	Pembagian	$x := 8 / 4;$
MOD	Sisa Pembagian	$x := 10 \text{ mod } 3;$

Contoh penulisan perhitungan aritmatika :

```
Hasil := 100 + 100;  
Diskon := (250000 * 10) / 100;
```

Contoh Program :

```
program keliling_persegi;  
uses crt;  
var  
    s, keliling : integer;  
  
begin  
    clrscr;  
    write('Masukan Nilai Sisi : '); readln(s);  
    keliling := 4 * s;  
    writeln('Keliling Persegi = ', keliling);  
    readln;  
end.
```



```
Masukan Nilai Sisi : 2  
Keliling Persegi = 8
```

Penjelasan : kita membuat proses input data agar user dapat memasukkan nilai sisi persegi, yang nantinya akan disimpan dalam variabel "s". Lalu kita lakukan perhitungan aritmatika sesuai rumus keliling persegi. Terakhir tampilkan hasil dari perhitungan keliling menggunakan perintah writeln.

3.6 Operasi Perbandingan

Dalam pemrograman pascal terdapat beberapa operator perbandingan yang digunakan untuk membandingkan dua buah data. Hasil dari perbandingan tersebut akan berupa nilai TRUE (benar) atau FALSE (salah).

Operator	Fungsi
=	Operasi sama dengan
>	Operasi lebih dari
<	Operasi kurang dari
>=	Operasi lebih dari atau sama dengan
<=	Operasi kurang dari atau sama dengan
<>	Operasi Tidak sama dengan

Operasi perbandingan seperti ini biasanya digunakan dalam struktur penyeleksian ataupun perulangan.

Contoh :

10 <= 7; (akan menghasilkan nilai TRUE)

11 > 20; (akan menghasilkan nilai FALSE)

Contoh Program :

```
program operasi_perbandingan;
uses crt;
var nama : string;

begin
  clrscr;
  nama := 'Khalid';
  writeln ( nama = 'Bilal' );
```

```
writeln ( nama <> 'Bilal');
readln;
end.
```

Penjelasan : Perbandingan “variable nama sama dengan Bilal” akan menghasilkan nilai false. Sedangkan perbandingan “variable nama tidak sama dengan Bilal” menghasilkan nilai true. Hasilnya seperti itu karena variabel *nama* berisi nilai ‘Khalid’.

3.7 Operasi Logika

Operasi logika juga akan menghasilkan nilai true atau false dari suatu perbandingan. Dalam buku ini hanya membahas 2 buah operator logika yang biasa digunakan dalam membuat program sederhana, yaitu : AND dan OR

AND akan menghasilkan nilai True jika semua data yang dibandingkan bernilai benar.

Contoh	Kondisi	Keterangan	Hasilnya
((2=2) AND (2>1))	((BENAR) AND (BENAR))	Semua data yang dibandingkan bernilai Benar	TRUE
((5<9) AND (2=2))	((SALAH) AND (BENAR))	Salah satu data yang dibandingkan bernilai Salah	FALSE
((3>5) AND (7<4))	((SALAH) AND (SALAH))	Semua data yang dibandingkan bernilai Salah	FALSE

Sedangkan **OR** akan menghasilkan nilai True jika salah satu atau semua data yang dibandingkan bernilai benar

Contoh	Kondisi	Keterangan	Hasilnya
((2=2) OR (2>1))	((BENAR) AND (BENAR))	Semua data yang dibandingkan bernilai Benar	TRUE

((5<9) OR (2=2))	((SALAH) AND (BENAR))	Salah satu data yang dibandingkan Bernilai Benar	TRUE
((3>5) OR (7<4))	((SALAH) AND (SALAH))	Semua data yang dibandingkan bernilai Salah	FALSE

3.8 Struktur Penyeleksian

Struktur Penyeleksian digunakan untuk menentukan perintah yang harus dijalankan pada kondisi tertentu.

Jika anda pernah melihat sebuah program yang dapat secara otomatis menentukan lulus/tidaknya nilai seorang mahasiswa, perlu anda ketahui bahwa program tersebut menggunakan struktur penyeleksian didalamnya.

Dalam pemrograman pascal terdapat 2 struktur perintah yang dapat digunakan untuk melakukan penyeleksian, yaitu "IF .. THEN" dan "CASE .. OF".

IF .. THEN

Bentuk umum penulisan : *IF kondisi THEN pernyataan;*

Jika *kondisi* terpenuhi (bernilai benar) maka *pernyataan* akan dijalankan, sedangkan jika *kondisi* tidak terpenuhi (bernilai salah) maka *pernyataan* tersebut tidak akan dijalankan dan program akan berlanjut ke perintah berikutnya setelah struktur "IF .. THEN".

Contoh Program :

```
program penyeleksian_if_1;
uses crt;
var a,b : integer;
begin
  clrscr;
```

```
  Nilai a lebih besar dari Nilai b
```

```
a:=10;
b:=5;
  if a >= b then writeln('Nilai a lebih besar dari Nilai b');
readln;
end.
```

Penjelasan: pada program diatas, Kondisinya adalah nilai $a \geq b$, apabila kondisi tersebut benar, maka text “Nilai a lebih besar dari Nilai b” akan ditampilkan.

Statement Block

Struktur penyeleksian yang memiliki lebih dari satu pernyataan yang akan dijalankan, maka harus ditambahkan “Statement Block” menggunakan perintah “**begin**” dan “**end;**” untuk mengelompokkan pernyataan-pernyataan yang akan dijalankan.

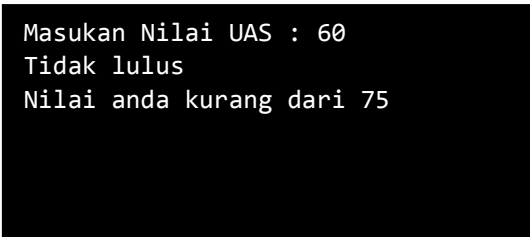
Penambahan statement block seperti ini juga berlaku pada struktur perintah yang lain seperti struktur perulangan.

Contoh penulisan statement block pada struktur IF :

```
IF kondisi Then
  Begin
    Pernyataan_1;
    Pernyataan_2;
    Pernyataan_3;
  End;
```

Contoh Program :

```
program penyeleksian_if_2;
uses crt;
var nilai:integer;
begin
  clrscr;
  write('Masukan Nilai UAS : ');readln(nilai);
  if nilai < 75 then
  begin
```



```
Masukan Nilai UAS : 60
Tidak lulus
Nilai anda kurang dari 75
```



```
writeln('tidak lulus');  
writeln('nilai anda kurang dari 75');  
end;  
readln;  
end.
```

Penjelasan : Pada program ini terdapat 2 pernyataan yang akan dijalankan saat kondisi terpenuhi, yaitu `writeln('tidak lulus');` dan `writeln('nilai anda kurang dari 75');`; maka diperlukan statement block untuk mengelompokkan 2 pernyataan tersebut.

IF .. THEN .. ELSE

Struktur IF..THEN dapat ditambahkan dengan perintah ELSE, fungsinya untuk dapat menentukan perintah yang harus dijalankan saat kondisi tidak terpenuhi (bernilai salah).

Bentuk umum penulisan :

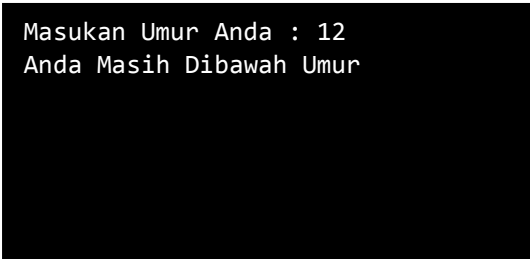
```
IF kondisi THEN  
    Pernyataan_1;  
ELSE  
    Pernyataan_2
```

Jika *kondisi* terpenuhi (benar) maka yang akan dijalankan adalah *pernyataan_1*, sedangkan jika kondisi tidak terpenuhi (salah) maka *pernyataan_1* akan diabaikan dan program akan menjalankan *pernyataan_2*.

Contoh Program :

```
program penyeleksian_if_3;  
uses crt;  
var umur:integer;  
begin  
    clrscr;
```

```
write('Masukan Umur Anda : ');readln(umur);
  if umur >= 18 then
    writeln('Anda Telah Dewasa')
  else
    writeln('Anda Masih Dibawah Umur');
readln;
end.
```



```
Masukan Umur Anda : 12
Anda Masih Dibawah Umur
```

CASE .. OF

Struktur case..of juga dapat digunakan untuk melakukan penyeleksian, namun hanya pada sebuah kondisi yang memiliki nilai sama (equivalent).

Bentuk umum Penulisan :

```
CASE ungkapan OF

  label_1 : pernyataan_1;

  label_2 : pernyataan_2;

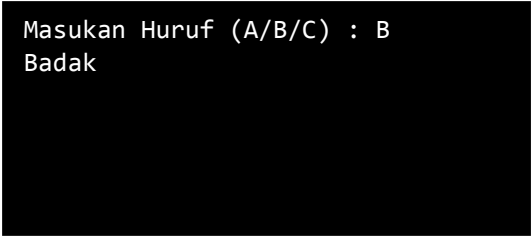
  label_3 : pernyataan_3;

end;
```

Saat *ungkapan* tersebut memiliki nilai yang sama (equivalent) dengan *label_1* maka yang akan dijalankan adalah *pernyataan_1*, begitu juga saat ungkapan tersebut bernilai sama dengan *label_2* maka yang akan dijalankan adalah pernyataan 2, begitu seterusnya.

Contoh program :

```
program penyeleksian_case_of;
uses crt;
var hewan:char;
begin
  clrscr;
  write('Masukan Huruf (A/B/C) : '); readln(hewan);
  case hewan of
    'A' : writeln ('Ayam');
    'B' : writeln ('Badak');
    'C' : writeln ('Cacing');
  end;
  readln;
end.
```



```
Masukan Huruf (A/B/C) : B
Badak
```

Penjelasan : anggaphlah ketika proses input anda memasukkan huruf 'B', maka ungkapan (variabel hewan) akan bernilai sama dengan label 'B', sehingga yang akan ditampilkan adalah 'Badak'.

CASE .. OF .. ELSE

Sama halnya dengan IF .. Then, perintah Case .. of juga dapat ditambahkan dengan perintah ELSE untuk menjalankan suatu pernyataan saat tidak ada kondisi yang terpenuhi.

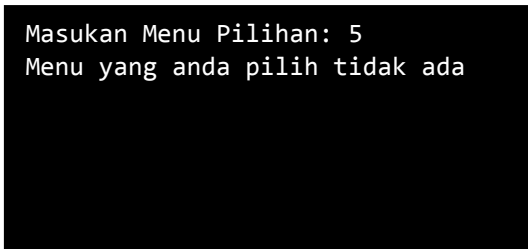
Bentuk umum penulisan :

```
CASE ungkapan OF
  label_1 : pernyataan_1;
  label_2 : pernyataan_2;
  label_3 : pernyataan_3;
ELSE
  pernyataan_4;
end;
```

saat nilai pada *ungkapan* tidak sama dengan *label_1* hingga *label_3* maka yang akan dijalankan adalah *pernyataan_4*

Contoh Program :

```
program penyeleksian_case_of_else;
uses crt;
var menu:integer;
begin
  clrscr;
  write('Masukan Menu Pilihan : ');readln(menu);
  case menu of
    1 : writeln ('Nasi Uduk');
    2 : writeln ('Telor Goreng');
    3 : writeln ('Pentol Kuah');
  else
    writeln ('menu yang anda pilih tidak ada');
  end;
end.
```



```
Masukan Menu Pilihan: 5
Menu yang anda pilih tidak ada
```

Penjelasan : saat program dijalankan dan anggaphlah pada proses input anda memasukkan angka 3 maka yang ditampilkan adalah text “Pentol Kuah”, tetapi jika yang anda masukkan adalah angka selain 1-3, maka yang akan ditampilkan adalah pernyataan yang ada pada bagian else.

Perbedaan dalam Menggunakan struktur IF .. Then dan Case .. OF

Struktur IF..Then dapat melakukan penyeleksian sesuai dengan kondisi yang anda buat, sedangkan case of hanya dapat melakukan penyeleksian pada kondisi yang bernilai *equivalent* terhadap label.

Terkadang dalam suatu kasus tertentu, membuat struktur penyeleksian akan menjadi lebih ringkas menggunakan CASE OF dari pada IF THEN

Silahkan anda lihat perbandingan struktur penyeleksian pada contoh kasus dibawah ini :

IF THEN	<pre>If no_kendaraan = 'DA' then daerah := 'Kalimantan selatan' Else if no_kendaraan = 'KH' then daerah := 'Kalimantan Tengah' Else if no_kendaraan = 'KT' then daerah := 'Kalimantan Timur' Else if no_kendaraan = 'B' then daerah := 'Jabodetabek' Else writeln ('No. Kendaraan Tidak ada');</pre>
CASE OF	<pre>Case no_kendaraan of 'DA' : daerah := 'Kalimantan Selatan'; 'KH' : daerah := 'Kalimantan Tengah'; 'KT' : daerah := 'Kalimantan Timur'; 'B' : daerah := 'Jabodetabek'; Else writeln ('No. Kendaraan Tidak ada');</pre>

Pada kasus diatas tentulah menggunakan case of akan lebih efisien dbandingkan menggunakan if..then.

3.9 Struktur Perulangan (Looping)

Struktur perulangan digunakan untuk mengulang suatu perintah sebanyak beberapa kali dalam kondisi tertentu. Perulangan akan sangat membantu saat menangani banyak proses yang harus dikerjakan secara berulang.

Contoh saja ketika anda harus membuat ratusan proses input data, tentu akan menyulitkan jika anda harus membuat ratusan perintah input secara manual. Maka anda perlu menggunakan struktur perulangan agar dapat membuatnya secara otomatis. Beberapa Perintah yang dapat menangani operasi perulangan dalam pemrograman pascal, yaitu "FOR .. DO", "WHILE .. DO", dan "REPEAT .. UNTIL".

FOR .. DO

Bentuk umum penulisan :

```
FOR counter := nilai_awal TO nilai_akhir DO pernyataan;
```

Awalnya, *counter* bernilai sama dengan *nilai_awal*. Saat program dijalankan, nilai *counter* tersebut akan terus bertambah (increase) sebanyak 1 nilai, dan *pernyataan* akan terus dijalankan secara berulang hingga jumlah nilai *counter* sama dengan *nilai_akhir*.

Contoh Program :

```
program perulangan_for;
uses crt;
var nomor:integer;
begin
    for nomor := 1 to 7 do writeln ('nomor ke ', nomor);
readln;
end.
```

```
Nomor ke 1
Nomor ke 2
Nomor ke 3
Nomor ke 4
Nomor ke 5
Nomor ke 6
Nomor ke 7
```

Penjelasan : saat dijalankan, nilai yang ada pada counter (variabel nomor) akan terus bertambah hingga mencapai angka 7 (nilai akhir). Sehingga perintah `writeln` akan terus diulang sebanyak 7 kali.

WHILE .. DO

Sama dengan `for`, hanya saja dengan menggunakan perintah `while do` anda dapat menyesuaikan kondisi serta menentukan nilai penambahan (increase) pada perulangan yang anda buat

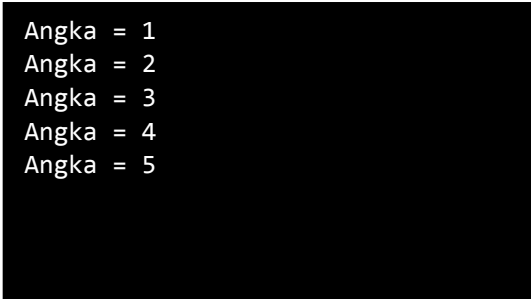
Bentuk umum penulisan :

```
While kondisi Do
    Pernyataan;
```

selama *kondisi* tersebut terpenuhi (benar) maka *pernyataan* akan terus dijalankan secara berulang. perulangan *pernyataan* ini akan otomatis berhenti saat *kondisi* tidak lagi terpenuhi (salah).

Contoh Program :

```
program perulangan_while;
uses crt;
var nomor:integer;
begin
clrscr;
    nomor := 1;
    while nomor <= 5 do
begin
    writeln ('angka = ', nomor);
    nomor := nomor + 1;
end;
readln;
end.
```

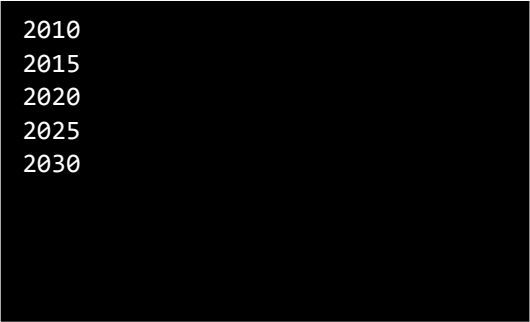


```
Angka = 1
Angka = 2
Angka = 3
Angka = 4
Angka = 5
```

Penjelasan : kita tentukan terlebih dahulu nilai awal dari counter (variabel nomor=1). pada program ini kondisinya adalah selama variabel nomor bernilai kurang dari 5 maka tulisan “Angka = ” Akan terus di tampilkan secara berulang. Nilai pada variabel nomor akan terus ditambahkan (increase) sebanyak 1 nilai, perulangan akan berhenti saat variabel nomor bernilai lebih dari 5 (kondisi sudah tidak terpenuhi lagi).

Contoh Program :

```
program perulangan_while_2;
uses crt;
var tahun:integer;
begin
clrscr;
    tahun := 2010;
    while tahun <= 2030 do
begin
    writeln (tahun);
    tahun := tahun + 5;
end;
readln;
end.
```



```
2010
2015
2020
2025
2030
```

Penjelasan : sama seperti contoh program sebelumnya, pertama kita menentukan nilai awal dari counter (variabel tahun=2010). pada program ini kondisinya adalah selama variabel tahun bernilai kurang dari 2030, maka variabel tahun Akan terus di tampilkan secara berulang.

Nilai pada variabel tahun akan terus ditambahkan sebanyak 5 nilai, perulangan akan dihentikan saat variabel tahun bernilai lebih dari 2030 (kondisi sudah tidak terpenuhi).

REPEAT .. UNTIL

Perulangan repeat until kebalikan dari perulangan while do. Jika dalam struktur while do perulangan akan dihentikan saat kondisi sudah tidak terpenuhi, maka dalam struktur repeat until perulangan akan berhenti ketika kondisi sudah terpenuhi.

Bentuk umum penulisan :

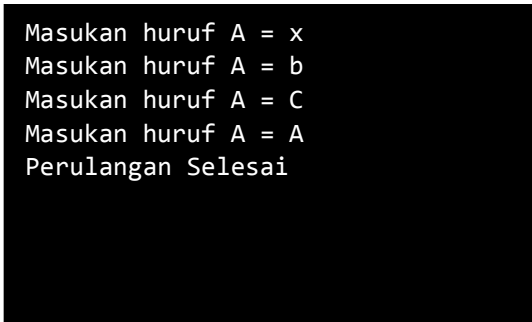
```
Repeat Pernyataan;
```

```
Until kondisi;
```

Pernyataan akan terus dijalankan secara berulang hingga *kondisi* tersebut terpenuhi (benilai benar).

Contoh program :

```
program perulangan_repeat;
uses crt;
var huruf:char;
begin
clrscr;
  repeat
    begin
      write('Masukan huruf A = ');readln(huruf);
    end;
  until huruf = 'A';
writeln('Perulangan Selesai')
readln;
end.
```



```
Masukan huruf A = x
Masukan huruf A = b
Masukan huruf A = C
Masukan huruf A = A
Perulangan Selesai
```


Penjelasan : proses input data akan terus diulang hingga kondisi terpenuhi. pada program ini kondisinya adalah variabel "huruf" sama dengan "A". Sehingga jika yang anda masukkan adalah selain dari karakter "A" maka proses input data akan terus berulang.

Struktur Perulangan Negative

Perulangan negative merupakan perulangan yang akan menghasilkan urutan secara terbalik dari besar ke kecil, perulangan negatif dapat dilakukan dengan perintah for ataupun while

FOR .. DO NEGATIF

Jika menggunakan perintah for, maka untuk menghasilkan perulangan negatif menggunakan struktur FOR .. DOWNTO .. DO

Bentuk umum penulisan :

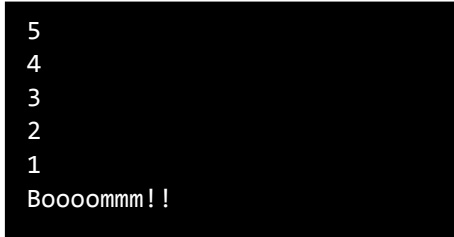
```
FOR counter := nilai_awal DOWNTO nilai_akhir DO pernyataan;
```

Counter akan terus berkurang nilainya hingga mencapai *nilai_akhir*.

Contoh Program :

```
program perulangan_for_negatif;
uses crt;
var timer:integer;

begin
  clrscr;
  for timer := 5 downto 1 do writeln (timer);
  writeln ('Boooooommm !!');
  readln;
end.
```



```
5
4
3
2
1
Boooooommm !!
```

Penjelasan : program ini akan melakukan perulangan dimulai dari angka 5 dan terus berkurang sebanyak 1 nilai hingga variabel timer bernilai 1. Setelah perulangan selesai text "BOOOM!" akan ditampilkan.

WHILE .. DO NEGATIF

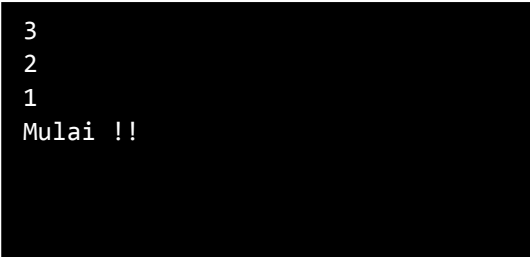
Membuat while do negatif yaitu dengan cara menyesuaikan kondisi serta pengurangan nilai (decrease) pada counter hingga menghasilkan perulangan negatif.

Bentuk umum penulisan :

```
While kondisi Do
  Begin
    Pernyataan;
    Counter := counter - pengurangan;
  end;
```

Contoh Program :

```
program perulangan_while_negatif;
uses crt;
var waktu:integer;
begin
  clrscr;
  waktu := 3;
  while waktu >= 1 do
  begin
    writeln (waktu);
    waktu := waktu - 1;
  end;
  writeln ('Mulai !!');
  readln;
end.
```



```
3
2
1
Mulai !!
```

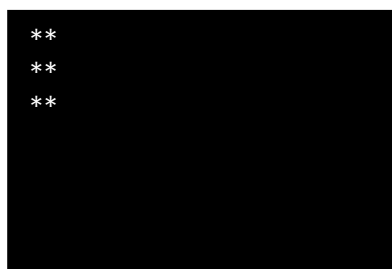
Penjelasan: program ini akan melakukan perulangan sebanyak 3 kali, variabel waktu dengan nilai awal 3 akan terus berkurang sebanyak 1 angka. Perulangan akan berhenti saat variabel waktu bernilai kurang dari atau sama dengan 1.

Struktur Perulangan Bersarang

Perulangan bersarang (nested loop) adalah perulangan yang memiliki struktur perulangan lagi didalamnya. Perulangan bersarang dapat menggunakan perintah for, while, ataupun repeat. Berikut ini adalah contoh perulangan bersarang menggunakan perintah for.

Contoh Program :

```
program perulangan_for_bersarang;
uses crt;
var ulang_atas,ulang_bawah:integer;
begin
  clrscr;
  for ulang_atas := 1 to 3 do
    begin
      for ulang_bawah := 1 to 2 do
        begin
          write('*');
        end;
        writeln;
      end;
    end;
  readln;
end.
```



Penjelasan : Cara kerja dari perulangan bersarang yaitu, perulangan yang diluar baru akan dijalankan setelah perulangan yang berada di dalamnya selesai.

Sehingga pada contoh program diatas Perulangan ulang_bawah akan terlebih dahulu dijalankan untuk menampilkan “*” sebanyak 2 kali, setelah itu barulah perulangan ulang_atas dijalankan untuk mengulang perulangan ulang_bawah sebanyak 3 kali hingga selesai. Dengan begitu akan membentuk “**” sebanyak 2 kolom dan 3 baris.

3.10 Array

Array termasuk dalam tipe data terstruktur, array adalah jenis tipe data yang dapat menampung lebih dari satu “element” dengan tipe data yang sama. Jika variabel biasa

hanya dapat menyimpan satu data didalamnya, maka dalam array dapat menyimpan lebih dari 1 data.

Bentuk umum penulisan :

```
var nama_array : array [rentang_index] of tipe_data;
```

Contoh deklarasi Array :

```
var buah : array [1..5] of string;
```

pada contoh diatas kita membuat sebuah array yang dapat menampung sebanyak 5 element data dengan tipe data string. untuk memudahkan anda dalam memahaminya, maka array dapat digambarkan dalam bentuk seperti ini :

Buah [1-5] : string

1	2	3	4	5

Masing-masing kotak kosong itu nantinya dapat di isi dengan suatu nilai yang memiliki tipe data string. Nomor diatasnya merupakan “index” yang merupakan nomor identitas dari setiap element yang ada.

Dalam pemrograman pascal sebuah array dapat membentuk rentang index hingga 255, artinya anda dapat membuat sebuah array yang dapat menampung maksimal 255 element data.

Memberikan nilai pada suatu Element Array

Sama halnya seperti varibel, setiap element array dapat ditentukan nilainya.

bentuk penulisan sebagai berikut :

```
nama_array [nomor_index] := isi_element;
```

Contoh penulisan :

```
Buah[4] := 'Anggur';
```

Artinya kita memberikan nilai “Anggur” pada array Buah index nomor 4, perlu anda ingat bahwa data yang diletakkan dalam element array harus sesuai dengan tipe data array tersebut.

Mengakses nilai pada element Array

untuk mengakses nilai yang ada pada suatu array dapat ditulis dalam bentuk :

```
nama_array[nomor_index];
```

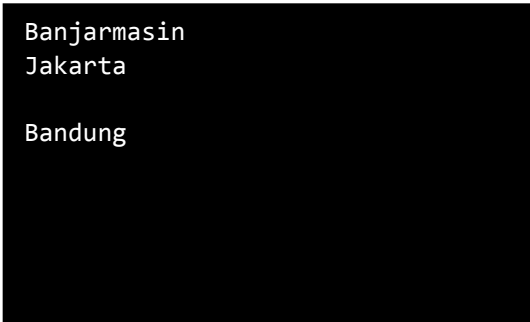
contoh untuk menampilkan isi array : `writeln(buah[4]);`

artinya kita akan menampilkan nilai yang ada pada array buah indeks nomor 4.

Contoh program :

```
program array_1;
uses crt;
var kota : array[1..5] of string;
begin
  clrscr;
  kota[1] := 'Banjarmasin';
  kota[2] := 'Jakarta';
  kota[4] := 'Bandung';

  writeln(kota[1]);
  writeln(kota[2]);
  writeln(kota[3]);
  writeln(kota[4]);
  writeln(kota[5]);
  readln;
end.
```



```
Banjarmasin
Jakarta

Bandung
```

Penjelasan : pertama kita deklarasikan sebuah array bernama kota yang memiliki 5 blok element dengan tipe data string. Selanjutnya array kota pada element ke 1,2, dan 4 kita isi dengan beberapa nama kota yang berbeda. Dan terakhir kita tampilkan data yang ada pada array kota element 1 hingga 5.

Jika digambarkan array pada contoh program tersebut dapat berbentuk seperti ini :

kota : string

1	2	3	4	5
Banjarmasin	Jakarta		Bandung	

Untuk mempermudah serta mempersingkat kode program biasanya array dikombinasikan dengan struktur perulangan.

Contoh program :

```

program array_perulangan;
uses crt;
var
    nomor, jumlah_data : integer;
    menu : array[1..20] of string;
begin
    clrscr;

    write ('Masukan Jumlah Pesanan : '); readln(jumlah_data);
    for nomor:=1 to jumlah_data do
    begin
        write('Masukan Pesanan ke ', nomor, ' : ');
        readln(menu[nomor]);
    end;
    Writeln;
    writeln('==== MENU YANG DIPESAN ====');
    For nomor:=1 to jumlah_data do
        writeln(nomor, ' . ', menu[nomor]);
    Writeln ('-----');
Readln;
end.

```

```

Masukan Jumlah Pesanan : 4
Masukan Pesanan ke 1 : Soto Ayam
Masukan Pesanan ke 2 : Nasi Goreng
Masukan Pesanan ke 3 : Mie Kuah
Masukan Pesanan ke 4 : Bakso

==== MENU YANG DIPESAN ====
1. Soto Ayam
2. Nasi Goreng
3. Mie Kuah
4. Bakso
-----

```

Penjelasan : kita buat proses input agar user dapat menentukan jumlah data pesanan. Pada baris selanjutnya kita buat perulangan untuk mengulang proses input sebanyak jumlah data yang telah ditentukan user dan akan disimpan dalam array "menu".

Terakhir kita buat satu buah perulangan lagi untuk menampilkan semua data yang telah di masukkan oleh user pada array “menu”.

3.11 Array 2 Dimensi

Array 2 Dimensi adalah array yang memiliki struktur element berupa baris dan kolom. Pada pembahasan array (1 dimensi) sebelumnya setiap elemen hanya memiliki 1 nomor indeks, sedangkan pada array 2 dimensi setiap elemen dapat memiliki 2 nomor indeks yang membentuk baris dan kolom.

Bentuk umum penulisan :

```
var nama_array:array[rentang_baris,rentang_kolom] of tipe_data;
```

contoh penulisan : `var minuman : array [1..2, 1..3] of string`

artinya kita membuat sebuah array dengan nama “minuman” yang memiliki 2 baris dan 3 kolom element.

Jika digambarkan dapat berbentuk seperti :

minuman [2 baris, 3 kolom] : string

	1	2	3
1			
2			

Memberikan nilai pada Element pada Array 2 Dimensi.

Untuk memberikan nilai pada pada element array 2 dimensi sama seperti sebelumnya hanya saja menggunakan 2 nomor index (baris dan kolom).

Bentuk umum penulisan :

```
nama_array[index_baris, index_kolom] := suatu_data;
```

Contoh penulisan :

```
minuman [1,2] := 'Teh Es';
```

```
minuman [2,3] := 'Jus';
```

Maka array minuman pada element baris 1 kolom 2 akan berisi nilai “Teh Es” Dan pada element baris 2 kolom 3 akan berisi nilai “Jus”.

minuman [2 baris – 3 kolom] : string

	1	2	3
1		Teh Es	
2			Jus

Mengakses nilai suatu Element pada Array 2 dimensi.

Bentuk umum Penulisan : `nama_array[index_baris, index_kolom];`

Contoh : `writeln (minuman[2,3]);`

Artinya kita akan menampilkan nilai dari element array minuman pada element baris 2 kolom 3.

Karena dapat membentuk baris dan kolom, Array 2 dimensi ini biasanya digunakan untuk membuat program matriks.

Contoh program :

```
program matriks_2_x_3;
uses crt;
var
  i,j: integer;
  matriks : array[1..2, 1..3] of string;
begin
  clrscr;
  {input nilai pada matriks}
  for i:=1 to 2 do
    begin
      for j:=1 to 3 do
```



```
begin
  write('nilai matriks baris-',i,' kolom-',j,': ');
  readln (matriks[i,j]);
end;
writeln;
end;

{tampilkan matriks}
writeln('Matriks 3 x 2 : ');
for i:=1 to 2 do
begin
  for j:=1 to 3 do
  begin
    write(matriks[i,j]:2);
  end;
  writeln;
end;
readln;
end.
```

```
nilai matriks baris-1 kolom-1: 4
nilai matriks baris-1 kolom-2: 3
nilai matriks baris-1 kolom-3: 1

nilai matriks baris-2 kolom-1: 4
nilai matriks baris-2 kolom-2: 2
nilai matriks baris-2 kolom-3: 5

Matriks 2 x 3 :
4 3 1
4 2 5
```

Penjelasan : dari contoh program diatas, Array 2 dimensi dikombinasikan dengan struktur Perulangan bersarang, sehingga dapat membentuk matriks dengan ordo 2x3. Nilai yang diinputkan dalam setiap element array akan ditampilkan menggunakan perulangan dengan struktur yang sama ketika proses input, sehingga posisi matriks saat ditampilkan dapat sesuai.

Tip : sebagai pengingat, dalam perulangan bersarang, proses pada perulangan yang berada didalam akan lebih dahulu dijalankan hingga selesai, barulah setelah itu perulangan diatasnya akan dijalankan.

3.12 Modular Programming

Modular Programming adalah suatu metode yang digunakan untuk menyusun struktur program. Dalam konsep modular programming, program utama akan dipecah kedalam

beberapa bagian (modul), setiap modul nantinya akan memiliki tugas atau fungsi masing-masing.

Contoh sederhananya, sebuah program kalkulator yang dipecah kedalam beberapa modul, yaitu modul A, B, C, dan D. Setiap modul tersebut nantinya akan menjalankan tugasnya masing-masing. Misal, modul A dibuat khusus untuk menangani proses penjumlahan, sedangkan proses pengurangan, perkalian, dan pembagian akan ditangani oleh modul yang berbeda.

Dalam bahasa pascal ada 2 perintah yang digunakan untuk menerapkan konsep modular programming yaitu Procedure dan Function.

Procedure

Procedure merupakan sub-program yang dibuat terpisah dari program utama. Nantinya, procedure yang dibuat akan dapat dijalankan dibagian manapun dalam program utama.

Deklarasi Procedure

Procedure dideklarasikan menggunakan perintah "procedure" yang diletakkan pada bagian header program. Instruksi yang akan dijalankan oleh procedure akan ditulis diantara **begin** dan **end;**

Bentuk umum penulisan :

```
procedure nama_procedure;  
begin  
  
end;
```

Dalam penggunaannya, procedure dapat diikuti oleh satu atau lebih variabel yang akan membawa suatu nilai kedalam procedure, atau disebut dengan *Parameter*.

Procedure yang menggunakan parameter dapat dideklarasikan dalam bentuk

```
procedure nama_procedure (parameter : tipe_data);  
begin  
  
end;
```

Menjalankan Procedure

Agar dapat dijalankan, procedure harus terlebih dahulu dipanggil melalui program utama. Procedure dapat dipanggil dengan menuliskan : `nama_procedure;`

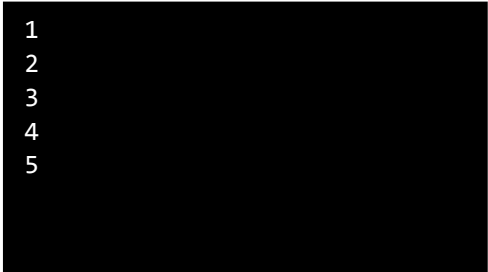
Procedure akan dijalankan secara berurutan dimulai dari procedure yang lebih dulu dipanggil. Tanpa adanya pemanggilan procedure maka Seluruh instruksi yang ada dalam procedure tidak akan dijalankan.

Contoh Program Procedure tanpa parameter :

```
program contoh_procedure;  
uses crt;  
var i : integer;
```

procedure berhitung;

```
begin  
  for i := 1 to 5 do writeln(i);  
end;  
  
begin  
  clrscr;  
  berhitung;  
  readln;  
end.
```



```
1  
2  
3  
4  
5
```


Penjelasan : pada contoh diatas, kita membuat sebuah procedure dengan nama “berhitung”. Procedure ini berisi struktur perulangan didalamnya, sehingga saat procedure “berhitung” dipanggil pada program utama, maka perulangan akan dijalankan.

Sedangkan jika anda tidak memanggil procedure tersebut kedalam program utama, maka seluruh perintah yang ada pada procedure tadi tidak akan dijalankan.

Contoh Program Procedure yang disertai parameter :

```
program contoh_procedure_paramenter;  
uses crt;  
  
procedure say_hello(nama : string);  
begin
```

```
writeln ('Hello, ', nama);  
end;  
  
begin  
  clrscr;  
  say_hello('Ahmed');  
  say_hello('Zakaria');  
  say_hello('Yahya');  
  readln;  
end.
```



```
Hello, Ahmed  
Hello, Zakaria  
Hello, Yahya
```

Penjelasan : pada contoh program diatas, kita membuat sebuah `procedure` `say_hello`; yang mana `procedure` ini memiliki sebuah “parameter nama” dengan tipe data string. Di bagian program utama kita panggil `procedure` `say_hello` diikuti dengan “parameter nama” yang berbeda.

Tip : anda dapat memanggil `Procedure` yang sama berkali-kali sesuai kebutuhan, sebagaimana contoh program diatas.

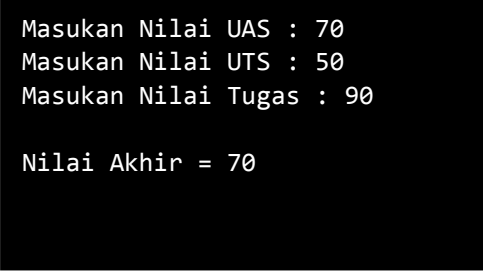
Contoh dibawah ini merupakan program menghitung nilai akhir yang dibuat menggunakan modular programming sederhana menggunakan `procedure`.

Contoh Program :

```
program contoh_procedure;  
uses crt;  
var uas, uts, tugas, nilai_akhir : real;  
  
procedure input_nilai;  
begin  
  write('Masukan Nilai UAS : ');readln(uas);  
  write('Masukan Nilai UTS : ');readln(uts);  
  write('Masukan Nilai Tugas : ');readln(tugas);  
  writeln;  
end;
```

```
procedure hitung_nilai_akhir;  
begin  
    nilai_akhir := (uas*0.4)+(uts*0.3)+(tugas*0.3);  
    writeln ('Nilai Akhir = ',nilai_akhir:0:0);  
end;
```

```
begin  
    clrscr;  
    input_nilai;  
    hitung_nilai_akhir;  
    readln;  
end.
```



```
Masukan Nilai UAS : 70  
Masukan Nilai UTS : 50  
Masukan Nilai Tugas : 90  
  
Nilai Akhir = 70
```

Penjelasan : Pada contoh program diatas, kita membuat procedure `input_nilai` yang bertugas untuk menangani proses input nilai dan procedure `hitung_nilai_akhir` yang bertugas untuk menghitung nilai akhir dari nilai-nilai yang telah dimasukan pada procedure input. Pada program utama kita panggil 2 buah procedure yang telah dibuat sebelumnya.

Function

Function juga merupakan sub-program, dan dalam penerapannya dapat diikuti oleh *parameter*. Perbedaannya dengan procedure adalah, Function harus dideklarasikan bersamaan dengan tipe data dari nilai yang dihasilkan oleh function tersebut.

Deklarasi Function

Sama halnya dengan procedure, function harus terlebih dahulu dideklarasikan pada bagian header program.

Bentuk umum penulisan :

```
function nama_function : tipe_data_function;  
begin  
end;
```

Function yang memiliki parameter dapat dideklarasikan dalam bentuk

```
function nama_function(paramenter:tipe_data):tipe_data_function;  
begin  
end;
```

Tip : sekedar mengingatkan, paramenter pada procedure maupun function hanya bersifat opsional, disesuaikan dengan kebutuhan. artinya anda boleh saja membuat procedure atau function tanpa disertai paramenter.

Menjalankan Function

Function dapat dijalankan dengan menuliskan : *nama_function* pada program utama.

Contoh Program function dengan paramenter :

```
program function_kalkulator;  
uses crt;
```

```
function penjumlahan(nilai1, nilai2 : integer) : integer;  
begin  
    penjumlahan := nilai1 + nilai2;  
end;
```

```
function pengurangan(nilai1, nilai2 : integer) : integer;  
begin  
    pengurangan := nilai1 - nilai2;  
end;
```

```
function pembagian(nilai1, nilai2 : integer) : real;  
begin  
    pembagian := nilai1 / nilai2;  
end;
```

```
function perkalian(nilai1, nilai2 : integer) : integer;  
begin  
    perkalian := nilai1 * nilai2;  
end;  
begin
```

```
clrscr;  
writeln (penjumlahan(11,2)); // 11 + 2  
writeln (perkalian(10,5)); // 10 x 5  
writeln (pengurangan(8,1)); // 8 - 1  
writeln (pembagian(9,3):0:0); // 9 / 3  
readln;  
end.
```

Penjelasan : Setiap function pada contoh diatas, dideklarasikan dengan tipe data integer, sesuai dengan nilai yang akan dihasilkannya. Masing-masing function akan menghasilkan nilai dari proses perhitungan yang terdapat dalam function tersebut. Setiap function juga memiliki 2 buah paramenter yang digunakan untuk mengirim nilai-nilai yang ingin diproses oleh masing-masing function.

Materi 4 Studi Kasus: Membuat Program Kasir Sederhana

Pada materi pembahasan terakhir ini, kita akan mencoba menerapkan seluruh materi yang telah dipelajari dari buku ini, dimulai dari Input & Output hingga Modular Programming kedalam sebuah kasus sederhana, yaitu membuat program KASIR.

Source code dari program Kasir ini dapat anda download melalui link yang tersedia pada halaman yang sama ketika anda mendownload buku ini.

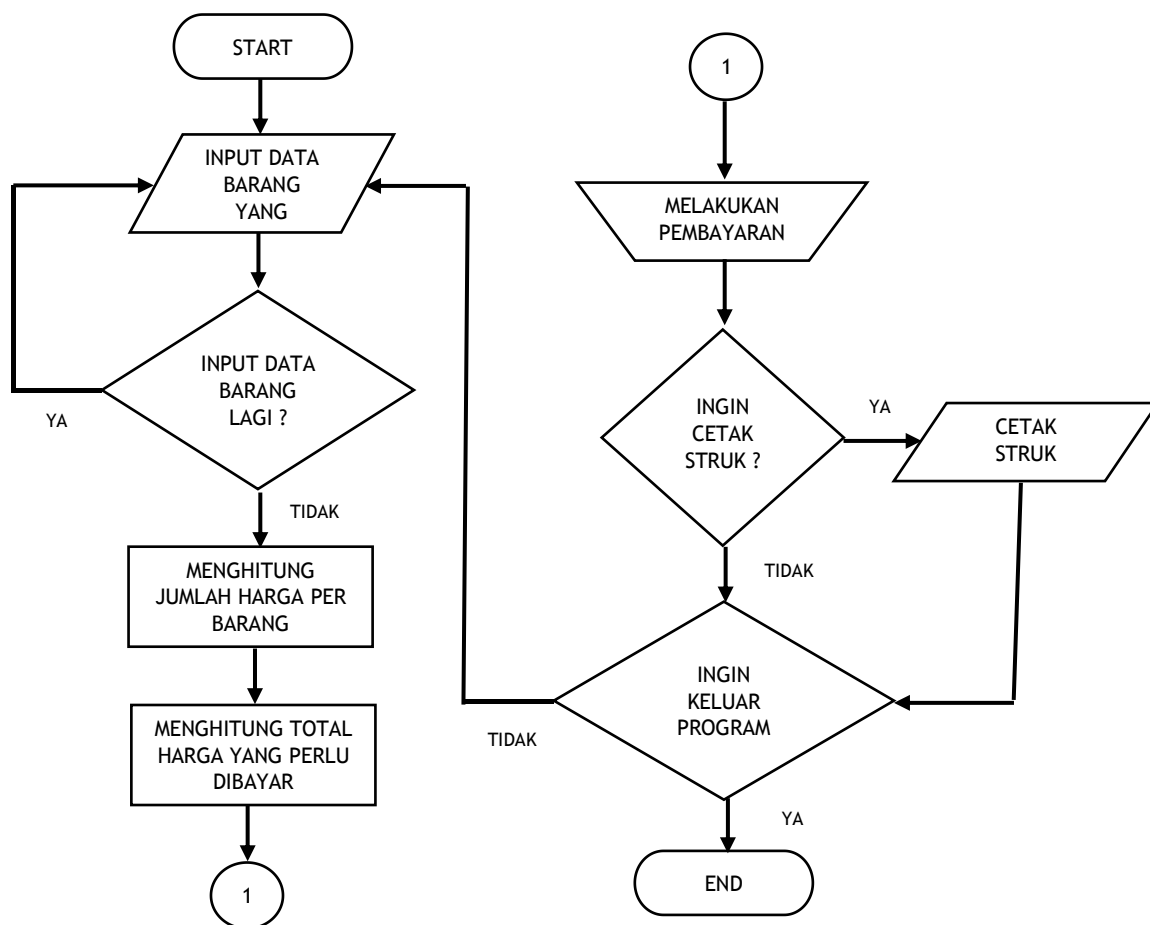
Adapun beberapa fitur yang terdapat dalam program Kasir ini, yaitu :

- Input pembelian, MAX 100 barang. Petugas kasir dapat memasukan data barang (nama barang, jumlah, dan harga satuan) yang dibeli sampai maksimal 100 barang.
- Menghitung Pembayaran & Kembalian Otomatis. Program dapat melakukan perhitungan total biaya pembelian yang perlu dibayar beserta uang kembalian secara otomatis.
- Menampilkan Struk pembelian. Petugas kasir dapat menampilkan struk pembelian berdasarkan data-data barang yang dibeli oleh pelanggan.

Sebelum menuliskan kode program pascal, silahkan anda perhatikan dan pahami terlebih dahulu Alur dari Program kasir yang akan dibuat, sehingga anda dapat membayangkan bagaimana proses jalannya program kasir tersebut.

Alur Program

Program kasir yang akan kita buat memiliki alur seperti yang ada pada flowchart dibawah ini. Setiap proses yang ada pada alur program nantinya akan kita pecah menjadi sebuah procedure.



- 1. Deklarasi Variabel**, Semua variable untuk data barang dibuat dalam bentuk array dengan rentang index 1-100, sehingga dapat menampung lebih dari 1 data barang dengan maksimal 100 data

```
program kasir_sederhana;
uses crt;
var
    jumlah_pembelian,i          : integer;
    cetak,keluar,input_lagi     : char;
    total_harga,kembalian,bayar : longint;

    nama_barang   : array[1..100] of string;
    harga_satuan  : array[1..100] of longint;
    jumlah_barang : array[1..100] of integer;
    jumlah_harga  : array[1..100] of longint;
```

- 2. Membuat Procedure input data barang**, Selanjutnya kita buat sebuah procedure dengan nama "input_barang_dibeli" yang bertugas untuk menangani proses input data barang (Nama Barang, Harga Satuan, dan Jumlah).

```
procedure input_barang_dibeli;
begin
    write('Nama Barang : ');
    readln(nama_barang[jumlah_pembelian]);
    write('Harga Satuan : ');
    readln(harga_satuan[jumlah_pembelian]);
    write('Jumlah      : ');
    readln(jumlah_barang[jumlah_pembelian]);
    writeln;
end;
```

nantinya procedure ini akan diletakkan didalam struktur perulangan, sehingga setiap data yang di inputkan dapat disimpan dalam masing-masing array berdasarkan nomor index yang disesuaikan dengan increase dari variable jumlah_pembelian.

- 3. Membuat Procedure Menghitung Jumlah Harga per Barang,** Buat juga sebuah procedure bernama “hitung_jumlah_harga” untuk menghitung jumlah harga dari barang yang di inputkan.

```
procedure hitung_jumlah_harga;
begin
    for i:=1 to jumlah_pembelian do
        begin
            jumlah_harga[i] := harga_satuan[i] * jumlah_barang[i];
        end;
    end;
end;
```

- 4. Membuat Procedure Menghitung Total Pembayaran,** Setelah itu buat procedure “hitung_total” yang bertugas untuk menghitung seluruh total pembayaran dengan menambahkan semua jumlah harga yang telah dihitung pada procedure hitung_jumlah_harga.

```
procedure hitung_total;
begin
    total_harga := 0;
    for i:=1 to jumlah_pembelian do
        begin
            total_harga := total_harga + jumlah_harga[i];
        end;
    end;
end;
```

- 5. Membuat Procedure Pembayaran,** procedure ini untuk menampilkan total harga yang perlu dibayar, serta memasukkan jumlah uang pembayaran kemudian menghitung jumlah uang kembalian secara otomatis.

```
procedure pembayaran;
begin
    clrscr;
    writeln ('+-----+');
    writeln ('|          APLIKASI KASIR SEDERHANA          |');
    writeln ('|                      PEMBAYARAN                      |');
```

```

writeln ('+-----+');
writeln;
writeln('Total Harga = ', total_harga);
write('Uang bayar  : ');readln(bayar);

kembalian := bayar - total_harga;

write('Kembalian  = ', kembalian);
end;

```

6. Membuat Procedure Untuk Mencetak Struk, saat procedure ini dijalankan maka akan menampilkan struk pembelian sesuai dengan data-data yang telah diproses pada procedure-procedure sebelumnya.

```

procedure cetak_struk;
begin
  clrscr;
  writeln ('    TOKO BUDI MAKMUR    ');
  writeln ('Jl. A. Yani KM 21 No. 10');
  writeln ('-----');

  for i:=1 to jumlah_pembelian-1 do
  begin
    writeln (nama_barang[i]);
    writeln ('    ',harga_satuan[i],' x ',jumlah_barang[i],' =
',jumlah_harga[i]);
  end;

  writeln('-----');
  writeln('Total = ', total_harga);
  writeln('Bayar = ',bayar);
  writeln('Kembalian = ',kembalian);
  writeln('-----');
  writeln('BARANG YANG TELAH DIIBELI');
  writeln('TIDAK DAPAT DIKEMBALIKAN');
  writeln('    TERIMAKASIH    ');

```

```
writeln('-----');
readln;
end;
```

7. Menyusun Program Utama, Setelah kita selesai membuat procedure yang dibutuhkan lengkap dengan tugasnya masing-masing. Langkah terakhir adalah menyusun program utama dengan menggabungkan semua procedure yang telah kita buat.

Procedure dipanggil secara berurutan sesuai dengan alur program, dimulai dari procedure `input_barang_dibeli`, `hitung_jumlah_harga`, `hitung_total`, `pembayaran`, hingga `cetak_struk`.

Perhatikan pada perulangan input barang, pada bagian tersebut, procedure “`input_barang_dibeli`” akan terus diulang dan nilai dari variable “`jumlah_pembelian`” akan terus bertambah hingga variable `input_lagi` bernilai = n (repeat until `input_lagi = 'n'`). Proses pada bagian ini nantinya akan membuat `jumlah_pembelian` sama dengan banyaknya jumlah barang yang di inputkan oleh petugas kasir. Saat variable `input_lagi` bernilai = 'n' maka proses akan dilanjutkan ke procedure berikutnya setelah procedure `input_barang_dibeli`.

```
begin
  keluar := 'n';
  repeat // perulangan seluruh sistem kasir selama, keluar = n
    begin
      clrscr;
      jumlah_pembelian:=1;
      input_lagi:='y';

      writeln ('+-----+');
      writeln ('|      APLIKASI KASIR SEDERHANA      |');
      writeln ('|              Versi 1.0              |');
      writeln ('+-----+');
      writeln;
      repeat // perulangan input barang
        begin
          input_barang_dibeli;
```

```
    jumlah_pembelian:=jumlah_pembelian+1;
    write('input Lagi ? (y/N) : ');input_lagi:=ReadKey;
    writeln;
    writeln('-----');
    writeln;
end;
until ((input_lagi='n') or (input_lagi='N'));

hitung_jumlah_harga;
hitung_total;
pembayaran;
writeln;
writeln;
write ('Cetak struk pembelian ? (Y/n) : ');
cetak := ReadKey;
if ((cetak = 'y') OR (cetak = 'Y')) then cetak_struk;
end;
writeln;
write ('Keluar dari Aplikasi Kasir ? : '); keluar:=ReadKey;
until ((keluar = 'y') or (keluar='Y'));
end.
```

Seluruh perintah yang ada dalam Program utama akan terus diulang hingga variabel “keluar” bernilai = y (repeat until keluar = ‘y’). Saat variabel keluar bernilai = y maka program KASIR akan dihentikan (end).

Sebagai info tambahan, jika kalian mengamati program utama diatas, ada beberapa proses inputan yang menggunakan perintah “ReadKey”. Fungsinya sama seperti “readln” hanya saja “ReadKey” dapat melakukan input sebuah karakter tanpa harus menekan tombol enter.

Jika anda berhasil menyusun kode program KASIR diatas dengan baik, saat dijalankan maka akan menampilkan tampilan sebagai berikut :

```

+-----+
|   APLIKASI KASIR SEDERHANA   |
|           Versi 1.0           |
+-----+

Nama Barang : RINSO MOLTO CAIR 800ML
Harga Satuan : 20000
Jumlah      : 1

input Lagi ? (y/N) :
-----

Nama Barang : HIT LILY BLOSSOM 275ML
Harga Satuan : 24000
Jumlah      : 1

input Lagi ? (y/N) :
-----

```

```

+-----+
|   APLIKASI KASIR SEDERHANA   |
|           PEMBAYARAN         |
+-----+

Total Harga = 83000
Uang bayar  : 100000
Kembalian   = 17000

Cetak struk pembelian ? (Y/n) :

```

```

          TOKO BUDI MAKMUR
          Jl. A. Yani KM 21 No. 10
          -----
          SUNLIGHT EXTRA REFF
            19000 x 1 = 19000
          RINSO MOLTO CAIR 800ML
            20000 x 1 = 20000
          HIT LILY BLOSSOM 275ML
            24000 x 1 = 24000
          BAYGON LAVENDER ISI 5PSG
            5000 x 4 = 20000
          -----
          Total = 83000
          Bayar = 100000
          Kembalian = 17000
          -----
          BARANG YANG TELAH DIIBELI
          TIDAK DAPAT DIKEMBALIKAN
          TERIMAKASIH
          -----

```