

MODUL AJAR INFORMATIKA

ALGORITMA DAN PEMMROGRAMAN (AP)

**DISUSUN OLEH :
ENDAH SUSANTI, S.KOM**

**TERDIRI DARI
9 AKTIVITAS PEMBELAJARAN**



```
PROGRAM ProcedureInputOutput;  
uses  
  crt;  
var  
  nama: string(25);  
begin  
  clrscr;  
  write('Masukkan nama Anda: ');  
  Readln(nama);  
  writeln('Nama Anda adalah ', nama);  
  writeln(10, 12); (memberikan bilangan 10 dan 12)  
  writeln(10, ' dan ', 11); (memberikan bilangan 10 dan 11)  
  writeln('jumlah= ', 100+50); (memberikan bilangan jumlah=150)  
  writeln('1416:0:2'); (memberikan bilangan 1.14)  
  Readln;  
end.
```

1 AKU DAN SEKOLAHKU

Nama saya Endah Susanti, Saya adalah guru mata pelajaran Informatika di SMA Negeri 2 Playen. Mengajar BTIK dan informatika. Selain melaksanakan membimbing dan mengajar peserta didik dan fasilitasi kepada pendidik dan tenaga kependidikan, saat ini saya mengemban amanah sebagai koordinator penilain dan koordinator tim IT sekolah. Mengabdikan sejak tahun 2003 sebagai pengajar ekstrakurikuler dan di tahun 2004 masuk sekolah sebagai guru TIK pada kurikulum KBK.

SMA Negeri 2 Playen adalah salah satu sekolah yang berada di Kabupaten Gunungkidul. Fasilitas untuk pembelajaran untuk Informatika cukup lengkap namun masih ada beberapa yang perlu dipenuhi. Fasilitas yang sudah ada sekarang adalah: jumlah komputer memenuhi jumlah siswa, untuk jaringan materi komputer sudah ada, koneksi internet sudah lancar dan memenuhi.

Waktu Pembelajaran untuk Bimbingan TIK tidak bisa diberikan secara maksimal, karena jika diberikan di jam ekstrakurikuler, pulanginya sudah sore, anak cenderung mengikuti ekstra wajib pramuka dan ekstra pilihan yang lain. Untuk tahun 2021 ini INFORMATIKA sudah masuk di pembelajaran lintas minat, jadi tidak semua siswa mendapatkan pembelajaran informatika di Kelas X, dan untuk jenjang berikutnya tetap masih bisa masuk dalam pembelajaran lintas minat.

Dari sisi intake Siswa, input siswa dari berbagai macam asal sekolah. Ada yang dari SMP nya sama sekali tidak diajarkan TIK dan ada juga sekolah yang sudah mengajarkan, ada anak yang benar-benar tidak bisa mengoperasikan aplikasi office seperti word, excel, powerpoint (padahal ini sangat penting untuk menunjang pembelajaran mata pelajaran yang lain). Bayangan anak ketika belajar informatika masih seputar belajar menggunakan aplikasi tidak pada menalar dan mencipta sebuah aplikasi/ produk.

Modul ajar ini disusun berdasarkan contoh program kecil dalam bahasa pascal yang disusun ibu Inggriani Liem Program Studi Informatia STIE- Isntitut Teknologi Bandung Versi: April 2007.

Dan tak lupa terimakasih kami sampaikan sebesar-besarnya kepada ibu Inggriani Liem yang telah banyak memberikan bahan program dari diktatnya, dan membantu penyusunan modul ajar ini, sehingga dapat tersusun dengan baik.

2 Acuan ATP

Capaian Pembelajaran :

<p>Pada akhir fase E, siswa mampu mengenal lebih dalam bagaimana komponen utama sistem komputer bekerja dan saling berinteraksi, enkripsi data, memahami internet dan jaringan lokal serta mengkoneksikan perangkat ke jaringan lokal dan internet, mengumpulkan dan mengintegrasikan data dari berbagai sumber baik secara manual atau otomatis menggunakan perkakas yang sesuai, mengintegrasikan potongan objek dalam berbagai format dari berbagai aplikasi untuk disajikan dalam berbagai representasi yang memudahkan analisis dan interpretasi, dan menggunakan fitur lanjut dan otomatisasi dari aplikasi perkantoran; b) mampu menerapkan berpikir komputasional dengan strategi algoritmik standar untuk mengembangkan program komputer yang terstruktur dalam bahasa pemrograman prosedural tekstual sebagai solusi atas persoalan berbagai bidang yang mengandung data diskrit bervolume tidak kecil, bergotong royong untuk</p>
--

menyelesaikan suatu persoalan kompleks dengan mengembangkan (merancang, mengimplementasi, memperbaiki, menguji) artefak komputasional yang bersentuhan bidang lain sesuai dengan kaidah proses rekayasa, serta mengkomunikasikan (mendokumentasi dan menjelaskan) rancangan produk, produk, dan prosesnya; dan c) mampu mengenal sejarah perkembangan komputer dan tokoh-tokohnya, memahami aspek teknis, hukum, ekonomi, lingkungan, dan sosial dari produk TIK, hak kekayaan intelektual, dan lisensi. mengenal berbagai bidang studi dan profesi terkait informatika serta peran informatika pada bidang lain

Capaian Pembelajaran Elemen Algoritma Pemrogram adalah:

Pada akhir fase E, siswa mampu memahami penerapan praktik baik konsep pemrograman prosedural dalam salah satu bahasa pemrograman prosedural dan mampu mengembangkan program yang terstruktur dalam notasi algoritma atau notasi lain, berdasarkan strategi algoritmik yang tepat.

Modul ini dikembangkan dengan memilih bahasa Pascal sebagai bahasa pemrograman prosedural tekstual.

3 Tujuan Pembelajaran

Kode TP	Tujuan Pembelajaran Elemen AP
X.AP.1	Membaca dan memahami algoritma dalam notasi algoritmik yang diajarkan di kelas
X.AP.2.	Menulis algoritma dengan notasi algoritmik yang diajarkan di kelas
X.AP.3.	Menjelaskan proses pemrograman di sebuah lingkungan perkakas bahasa pemrograman prosedural tekstual
X.AP.4.	Menulis program prosedural tekstual dengan struktur yang benar
X.AP.5.	Menjelaskan input dan output dalam struktur program prosedural tekstual
X.AP.6.	Membuat program yang dapat membaca input dan menampilkan output
X.AP.7.	Menjelaskan arti Type, variabel, const dan ekspresi dalam program
X.AP.8.	Mengidentifikasi penulisan variabel yang benar dalam suatu bahasa pemrograman
X.AP.9.	Membuat program yang mempergunakan variabel dan ekspresi
X.AP.10.	Merancang dan membuat program prosedural tekstual yang memuat struktur kontrol kondisional
X.AP.11	Merancang dan membuat program prosedural tekstual yang memuat struktur kontrol perulangan
X.AP.12.	Mengimplementasi penggunaan array dalam penyelesaian persoalan pemrograman
X.AP.13.	Mengimplementasi penggunaan fungsi dalam penyelesaian persoalan pemrograman
X.AP.14	Menghasilkan solusi Permasalahan dan menghasilkan solusi dalam bentuk program prosedural tekstual

4 Identitas Modul

Nama	Endah Susanti, S.Kom	Jenjang/Kelas	SMA/ X	INF.C.ENS.10.AP- X1
Asal sekolah	SMA Negeri 2 Playen	Mapel	Informatika	
Alokasi waktu	9 x pertemuan (27 JP) 1215 menit	Jumlah siswa	Maksimal 36 siswa	
Profil pelajar Pancasila yang berkaitan	<ul style="list-style-type: none">● Mandiri● Kreatif● Bernalar kritis● Bergotong royong	Model pembelajaran	<ul style="list-style-type: none">▪ Tatap muka▪ Daring	
Fase	E	Domain Mapel	Algoritma dan Pemrograman	
Tujuan Pembelajaran	X.AP.1. Membaca dan memahami algoritma dalam notasi algoritmik yang diajarkan di kelas X.AP.2. Menulis algoritma dengan notasi algoritmik yang diajarkan di kelas X.AP.3. Menjelaskan proses pemrograman di sebuah lingkungan perkakas bahasa pemrograman prosedural tekstual X.AP.4. Menulis program prosedural tekstual dengan struktur yang benar X.AP.5. Menjelaskan input dan output dalam struktur program prosedural tekstual X.AP.6. Membuat program yang dapat membaca input dan menampilkan output X.AP.7. Menjelaskan arti Type, variabel, const dan ekspresi dalam program X.AP.8. Mengidentifikasi penulisan variabel yang benar dalam suatu bahasa pemrograman X.AP.9. Membuat program yang mempergunakan variabel dan ekspresi X.AP.10. Merancang dan membuat program prosedural tekstual yang memuat struktur kontrol kondisional X.AP.11 Merancang dan membuat program prosedural tekstual yang memuat struktur kontrol perulangan X.AP.12. Mengimplementasikan penggunaan array dalam penyelesaian persoalan pemrograman X.AP.13.Mengimplementasi penggunaan fungsi dalam penyelesaian persoalan pemrograman X.AP.14. Menghasilkan solusi Permasalahan dan menghasilkan solusi dalam bentuk program prosedural tekstual			
Kata kunci	Koding (<i>coding</i>), Pemrograman (<i>programming</i>), <i>procedural programming</i> , bahasa pemrograman Pascal, praktik baik pemrograman, <i>problem solving</i> , program reading, program <i>comprehension</i> , program <i>writing</i>			
Deskripsi umum kegiatan	Secara umum, kegiatan yang akan dilakukan siswa dalam modul AP ini adalah: (1) memahami struktur dasar program prosedural dalam bahasa pascal melalui kegiatan koding sesuai praktik baik berdasarkan contoh program kecil yang merupakan pola program. Kegiatan diisi dengan latihan “coding” mengetikkan contoh program kecil yang diberikan.			

	<p>Siswa melakukan proses “coding” secara mandiri maupun “pairing” yaitu melakukan dalam kelompok yang terdiri dari dua orang. Satu siswa mengetik, dan satu siswa mengamati serta menguji program untuk memastikan benar. Peran sebagai pengetik dan penguji harus ditukar untuk program yang berbeda.</p> <p>(2) problem solving masalah sederhana dengan solusi sebuah program pascal.</p>
Materi ajar, alat, dan bahan	<p>Materi Ajar : <i>Free Pascal</i></p> <p>Alat dan bahan :</p> <ul style="list-style-type: none"> ▪ Lembar kerja siswa dicetak atau (menggunakan Google Docs/Form)
Sarana Prasarana	<p>Tatap Muka: papan tulis, lembar kerja, komputer</p> <p>PJJ Daring : Google meet, whatsapp, classroom</p>
Target Peserta Didik	Siswa Regular
Ketersediaan Materi	<ol style="list-style-type: none"> 1. Materi untuk melakukan kegiatan pembelajaran selama 9 pertemuan 2. Lembar Kerja Siswa setiap aktivitas 3. Materi Pengayaan secara global
Kegiatan pembelajaran Utama	<p>Pengaturan siswa: bekerja pairing untuk latihan Coding, bekerja kelompok untuk mini project</p> <p>Metode :</p> <ul style="list-style-type: none"> ▪ Pertemuan 1 : Diskusi, instalasi perkakas pemrograman ▪ Pertemuan 2 : <i>Coding, program comprehension</i> ▪ Pertemuan 3 : Praktik, <i>problem Solving</i> ▪ Pertemuan 4 : Praktik, <i>problem solving</i> ▪ Pertemuan 5 : Praktik, <i>problem solving</i> ▪ Pertemuan 6 : praktik, <i>problem solving</i> ▪ Pertemuan 7 : <i>Coding, program comprehension</i> ▪ Pertemuan 8 : <i>Coding, program comprehension</i> 1. ▪ Pertemuan 9 : mini project, <i>problem solving</i>
Asesmen	<ul style="list-style-type: none"> ▪ Reading comprehension dan Coding Program Kecil ▪ Problem Solving
Persiapan Pembelajaran	<ul style="list-style-type: none"> ▪ Guru mempersiapkan dan memiliki modul ajar yang akan diajarkan ▪ Guru mempersiapkan lembar kerja siswa dan penilaian (asesmen). Guru mempersiapkan diri dengan mencoba contoh program kecil yang akan digunakan sebagai latihan dan LKS, minimal sehari sebelum pembelajaran

5 Kontribusi ke Profil Pelajar Pancasila dan Core Practices PLB

Kegiatan dan Pengalaman Bermakna	Profil Pancasila	Berpikir Komputasional	Praktik AP
<p>Mengenali dan mendefinisikan persoalan yang pemecahannya dapat didukung dengan sistem komputasi;</p> <p>Mengembangkan dan menggunakan abstraksi untuk memodelkan masalah;</p> <p>Mengembangkan artefak komputasi dengan membuat desain program sederhana untuk menunjang model komputasi yang dibutuhkan di pelajaran lain;</p> <p>Mengembangkan rencana pengujian, menguji dan mendokumentasikan hasilnya;</p>	<p>Mandiri</p> <p>Bernalar kritis</p> <p>Kreatif</p>	<p>Abstraksi</p> <p>Dekomposisi</p> <p>Algoritma</p> <p>Pengenalan Pola</p>	<p>Program reading/ comprehension lewat program contoh</p> <p>Coding</p> <p>Program debugging & Testing</p> <p>Algorithm/Program design</p> <p>Problem solving</p>
<p>Mengkomunikasikan suatu proses, fenomena, solusi TIK dengan mempresentasikan, memvisualisasikan serta memperhatikan hak kekayaan intelektual;</p> <p>Memiliki budaya kerja masyarakat digital dalam tim yang inklusif;</p> <p>Berkolaborasi untuk melaksanakan tugas dengan tema komputasi;</p>	<p>Bernalar Kritis</p> <p>Bergotong royong</p>	<p>Abstraksi</p> <p>Dekomposisi</p>	<p>Penerapan Best practices dalam programming</p> <p>Demo program</p> <p>Bekerja pairing</p> <p>Bekerja dalam kelompok</p>

6 Konsep Utama

Modul ajar ini mencakup materi yang tergambar dalam dekomposisi materi Algoritma Pemrograman yang tersaji dalam diagram hirarkis sebagai berikut :



6.1

Apersepsi – membawa anak dari dunia nyata ke dunia pemrograman

Sebuah restoran di Singapore menyediakan robot yang akan memasak makanan, menggantikan koki, dapat menyediakan nasi goreng dalam 20 detik. Tahukah kamu bahwa proses memasak nasi goreng yang dikerjakan oleh robot tersebut adalah sebuah program yang “ditanamkan” dalam mesin ?

<https://www.dailymail.co.uk/femail/food/article-9024959/Singapore-cafe-Bowl-Bowl-features-ROBOTS-cook-fried-rice-just-20-seconds.html>

6.2 Pemetaan Tujuan-Konsep-Pertemuan-Aktivitas

Pada bagian ini, Unit pembelajaran dipetakan menjadi aktivitas konkrit

Tujuan Spesifik Pembelajaran	Topik/Konsep	Kode-Aktivitas	Plugged/ Unplugged	Pertemuan ke-	Jam (JP)
X.AP.1. Membaca dan memahami algoritma dalam notasi algoritmik yang diajarkan di kelas	Flowchart dan notasi algoritmik	K10-AP-U1-A1-Flowchart	Unplugged	1	3

X.AP.2. Menulis algoritma dengan notasi algoritmik yang diajarkan di kelas					
X.AP.3. Menjelaskan proses pemrograman di sebuah lingkungan perkakas bahasa pemrograman prosedural tekstual	Pengenalan IDE dan Koding di lingkungan IDE, Code Convention	K10-AP-P1-A2-IDE Pascal	Plugged	2	3
X.AP.4. Menulis program prosedural tekstual dengan struktur yang benar					
X.AP.5. Menjelaskan input dan output dalam struktur program prosedural tekstual	Input, output program pascal Tipe data dasar, Operator aritmatika : {+, -, *, div, /, ^}	K10-AP-P2-A3-Input Output	Plugged	3	3
X.AP.6. Membuat program yang dapat membaca input dan menampilkan output					
X.AP.7. Menjelaskan arti Type, variabel, const dan ekspresi dalam program	Type, variabel, const dan ekspresi dalam program	K10-AP-P3-A4-Assign	Plugged	4	3
X.AP.8. Mengidentifikasi penulisan variabel yang benar dalam suatu bahasa pemrograman					
X.AP.9. Membuat program yang mempergunakan variabel dan ekspresi					
X.AP.10. Merancang dan membuat program prosedural tekstual yang memuat struktur kontrol kondisional	Struktur kontrol keputusan (kondisional)	K10-AP-P4-A5-Analisis kondisi	Plugged	5	3
X.AP.11. Merancang dan membuat program prosedural tekstual yang memuat struktur kontrol perulangan	Struktur kontrol perulangan	K10-AP-P5-A6-loop	Plugged	6	3

X.AP.12.	Mengimplementasikan penggunaan array dalam penyelesaian persoalan pemrograman	Array	K10-AP-P6-A7-array	Plugged	7	3
X.AP.13.	Mengimplementasikan penggunaan subprogram dalam penyelesaian persoalan pemrograman	Function dan prosedur	K10-AP-P7-A8-subprogram	Plugged	8	3
X.AP.14.	Menghasilkan solusi Permasalahan dan menghasilkan solusi dalam bentuk program prosedural tekstual	Mini project tentang temperatur air	K10-AP-P8-A9-miniproject	Plugged	9	3

Bahasa pemrograman yang dipilih dalam modul ajar ini AP fase E ini adalah bahasa Pascal karena merupakan bahasa pemrograman prosedural tekstual yang sederhana sintaknya. Pada aktivitas dengan kode aktivitas K10-AP-P11-A2-IDE Pascal sampai dengan kode aktivitas K10-AP-P7-A8-subprogram, siswa akan belajar memahami struktur program dan koding dalam bahasa PASCAL lewat sejumlah contoh program kecil. Program kecil ini akan menjadi bahan belajar program comprehension (baik dari segi teksnya, maupun eksekusinya), dan sekaligus mengembangkan keterampilan koding. Setiap program kecil akan mewakili satu konsep pemrograman prosedural, dan merupakan program utuh yang siap dieksekusi, menjadi bahan belajar struktur program dan eksekusinya. Kumpulan program kecil akan menjadi pola solusi berupa kode program, yang akan dipakai untuk menyelesaikan persoalan-persoalan yang lebih besar.

Sedangkan pada K10-AP-P8-A9 siswa akan melakukan problem solving untuk kasus kecil dan sangat sederhana, yang solusinya berupa program. Pada mini project inilah siswa akan melakukan programming (tidak hanya koding). Keseluruhan aktivitas diperkirakan akan memerlukan 9 minggu.

Satu seri program kecil yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pemecahan masalah sederhana yang memuat semua konsep pemrograman prosedural yang diharapkan. Setiap program kecil juga ditulis dalam kaidah pengkodean yang mengikuti praktek baik :

1. Penamaan dan deklarasi variabel dan konstanta:
 - a. Nama variabel atau konstanta harus sesuai dengan aturan bahasa pemrograman yang dipakai (dalam hal ini bahasa Pascal) dan bermakna.
 - b. Jika nama sangat singkat harus dijelaskan maknanya dengan komentar.
 - c. Nilai konstanta harus sesuai dengan namanya.
2. Program mengandung penjelasan dalam bentuk komentar, sehingga memudahkan dibaca oleh manusia.
3. Program ditulis dengan indentasi yang baik dan seragam.

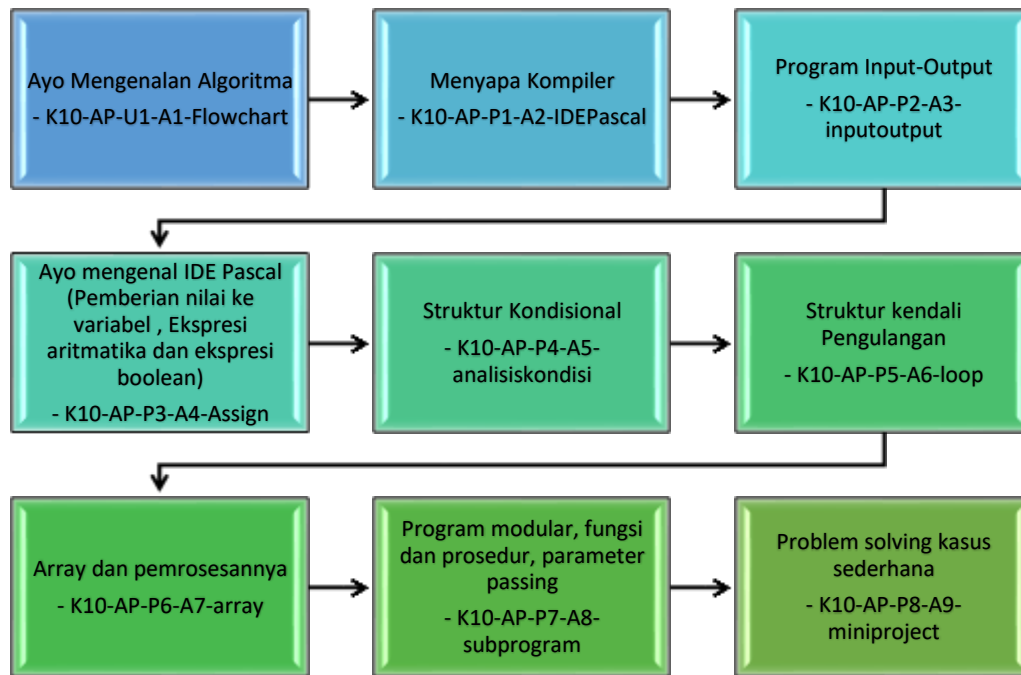
Jika ingin mendalami aturan pengkodean yang baik, karena kita menggunakan free pascal, maka dapat mengacu ke https://wiki.freepascal.org/Coding_style

Setelah memahami translasi notasi algoritmik ke bahasa Pascal, dan memahami konsep translasi dari bahasa yang mirip, siswa dapat belajar mentranslasi program Pascal menjadi program dalam bahasa C, dengan mengacu ke aturan translasi, yang misalnya dapat diakses di <http://psy.swansea.ac.uk/staff/carter/projects/Pas2C.htm>

Daftar Program kecil, konsep yang diperlukan dan pertemuannya diberikan dalam tabel sebagai berikut :

Konsep	Contoh Program	Pertemuan
Menyapa kompiler	hello.pas hellodos.pas	2
Input-output	baca.pas tabel.pas tuliswrite.pas penjumlahan.pas	
Pemberian nilai ke variabel	assign.pas assign1.pas deklarasi.pas deklarasi1.pas	3
Ekspresi aritmatika dan ekspresi boolean	oprator.pas tulisnilai.pas	4
Struktur kendali kondisional	if1.pas if2.pas if3.pas kasus.pas	5
Struktur kendali Pengulangan	fordo.pas downto.pas whiledo.pas repeatuntil.pas bintang.pas	6
Array dan pemrosesannya	tabel.pas array.pas searcharray.pas	7
Program modular, fungsi dan prosedur, parameter passing	subprg.pas lingkup.pas funcrec.pas	8
Problem solving kasus sederhana	temperatur.pas	9

Graph ketergantungan aktivitas :



7 Deskripsi Aktivitas

7.1 Aktivitas – Ayo Mengenal Algoritma Kode Aktivitas : K10-AP-U1-A1-Flowchart

Pada aktivitas pembelajaran ini siswa akan di kenalkan dengan apa itu algoritma dan sampai pada membaca dan menulis algoritma.

7.1.1 Pertanyaan Pemantik

Bagaimana penyusunan jadwal siswa masuk pada kegiatan belajar di era Newnormal dengan menggunakan nomor kelompok ganjil-genap ?

7.1.2 Konsep Terkait Aktivitas

Algoritma adalah urutan prosedur berupa langkah-langkah yang akan dilakukan untuk memecahkan masalah.

Algoritma memiliki aturan sendiri dalam penulisannya yang disebut dengan **Notasi Algoritma**. Notasi Algoritmik selalu terdiri dari 3 bagian yaitu:

1. Judul (Header), adalah bagian teks algoritma yang berfungsi untuk mendefinisikan apakah teks tersebut adalah program, prosedur, fungsi, modul atau sebuah skema program
2. Deklarasi/Kamus, bagian teks algoritma yang digunakan untuk mendefinisikan nama type, nama konstanta, nama variabel, nama fungsi, nama prosedur
3. Algoritma, adalah bagian teks algoritmik yang berisi instruksi atau pemanggilan aksi yang telah didefinisikan. Komponen teks algoritmik dalam pemrograman procedural dapat berupa:
 - a. Instruksi dasar seperti input/output, assignment
 - b. Sequence

- c. Analisa kasus
- d. Pengulangan

Notasi algoritma ini tidak tergantung pada spesifikasi bahasa pemrograman tertentu. Untuk membuat algoritma dari sebuah permasalahan yang akan di tuangkan dalam pemrograman biasanya dapat menggunakan 3 jenis notasi algoritma yaitu : menggunakan bahasa sehari-hari, flowchart (diagram alir) atau *pseudocode*.

PENULISAN ALGORITAMA

Sebelum ditulis dalam bahasa pemrograman, solusi suatu persoalan dapat dituliskan dalam beberapa cara, yaitu sebagai berikut :

1. Menggunakan bahasa sehari-hari berupa teks yang terstruktur, yang dituangkan dalam urutan langkah.
2. Menggunakan **Flowchart**, yaitu ditulis dengan simbol-simbol yang mewakili urutan instruksi pengambilan keputusan, pengulangan, atau uraian kejadian pemecahan masalah.
3. Menggunakan **Pseudocode**, yaitu dituliskan mendekati perintah bahasa pemrograman yang akan digunakan sebagai alat implementasi program.

Ciri penting algoritma

1. Algoritma harus berhenti setelah mengerjakan sejumlah langkah terbatas.
2. Setiap langkah harus didefinisikan dengan tepat dan tidak berarti-dua (Ambiguitas).
3. Algoritma memiliki nol atau lebih masukan.
4. Algoritma memiliki nol atau lebih keluaran.
5. Algoritma harus efektif (setiap langkah harus sederhana sehingga dapat dikerjakan dalam waktu yang masuk akal).

A. FLOWCHART

Flowchart / diagram alir terdiri dari sekumpulan simbol dimana setiap simbol menggambarkan sebuah arti tertentu. Flowchart ini lebih banyak digunakan karena bentuknya yang sederhana dan memberikan gambaran alur visual. Tetapi flowchart sangat menyulitkan untuk program yang panjang dan rumit karena menjadi sulit untuk dibaca. Flowchart juga lebih mudah untuk memahami alur eksekusi, yang lebih sulit diabstraksikan ketimbang *pseudocode*.

A.1. Fungsi Flowchart

Flowchart digunakan untuk menganalisis, mendesain, mendokumentasikan sebuah proses atau program di berbagai bidang, bukan hanya program komputer. Secara khusus, flowchart berfungsi untuk membantu menggambarkan proses apa yang sedang terjadi dan yang akan terjadi dari sebuah simbol dan tanda penghubungnya. Selain itu, flowchart ini mampu memperjelas sebuah alur dari suatu sistem.

Dalam konteks modul ini, flowchart dipakai untuk mendeskripsikan alur eksekusi sebuah program.

A.2. Tujuan Flowchart




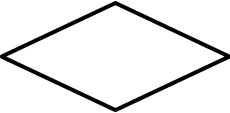
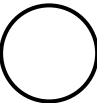


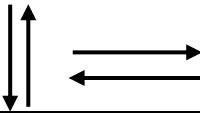
Flowchart dibuat untuk: (a) menggambarkan urutan atau tahapan dari penyelesaian masalah; (b) alur eksekusi sebuah program

Flowchart membantu analis dan pemrogram untuk menuangkan solusi persoalan ke dalam bagian-bagian yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian apa.

A.3. Siklus dalam Flowchart

Siklus dalam flowchart adalah siklus input-proses-output atau sering disebut siklus IPO. Siklus IPO dapat kita analogikan sebagai komponen hardware komputer. Komponen input seperti keyboard, mouse, scanner, dan sebagainya dapat kita andaikan sebagai input (penerimaan suatu perintah atau data yang akan diproses).

A.4. Simbol Flowchart

NO	LAMBANG	NAMA	FUNGSI
1		Terminal	Menyatakan permulaan atau akhir suatu program
2		Input/ output	Mnyatakan proses input/ output tanpa tergantung jenis peralatannya
3		Process	Menyatakan suatu proses yang dilakukan oleh komputer
4		Decision	Menunjukkan suatu kondisi tertentu yang akan menghasilkan beberapa kemungkinan : ya atau tidak, atau kondisi lain yang dituliskan dalam alur percabangan keluarnya.
5		Connector	Menyatakan sambungan dari proses ke proses lainnya dalam halaman yang sama
6		Predefined Process (subprogram)	Menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal
7		Document	Mencetak keluaran dalam bentuk dokumen (melalui printer
8		Flow	Menyatakan alur suatu proses atau instruksi dari satu simbol ke simbol lainnya

B. PSEUDOCODE

Sedangkan pseudocode adalah algoritma yang betuknya sangat mirip dengan bahasa pemrograman. Dalam penulisannya terdiri dari 3 bagian yaitu : Judul Algoritma, Deklarasi, Deskripsi . Deskripsi dituliskan dalam kalimat terstruktur dengan pola yang sudah ditentukan,

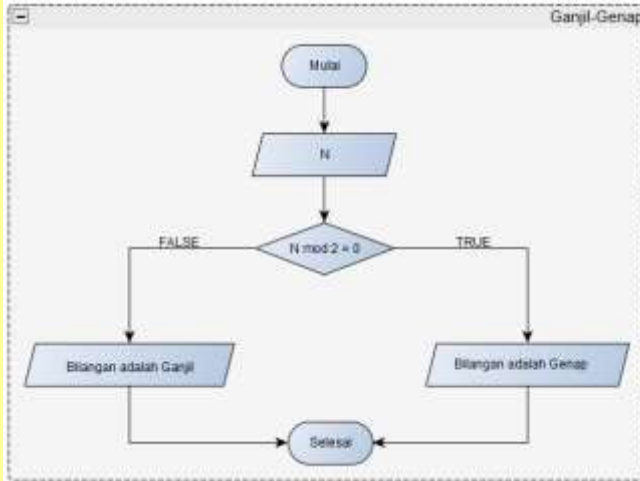
yang menunjukkan elemen pemrograman yaitu : assignment, kondisional, pengulangan. Teks dalam pseudocode dituliskan terindentasi.

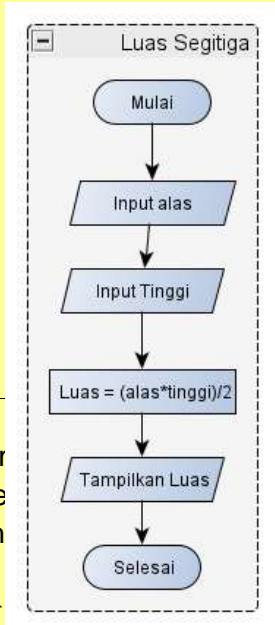
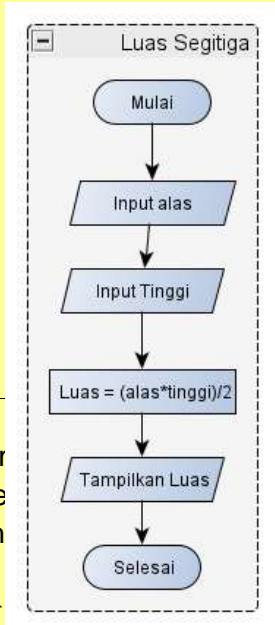
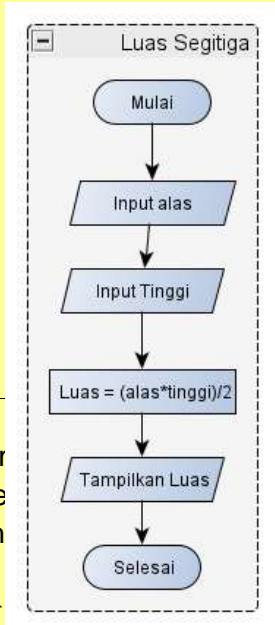
7.1.3 Kata kunci

Algoritma, Flowchart, notasi algoritmik, membaca algoritma, menulis algoritma

7.1.4 Gambaran Umum Kegiatan

KEGIATAN	DESKRIPSI KEGIATAN
Pendahuluan	<p>Diawal pembelajaran guru memberikan sebuah permasalahan komputasi sederhana sebagai umpan diskusi yaitu : tentukan langkah yang tepat untuk menghasilkan solusi algoritmik untuk persoalan yang diberikan, yaitu seperti berikut : Pada perioda new normal, ada pembatasan jumlah peserta kelas sehingga siswa digilir untuk belajar tatap muka. Pak Guru memutuskan untuk membagi siswa menjadi dua kelompok, berdasarkan nomor siswanya yaitu nomor ganjil dan nomor genap. Andaikata siswa akan punya sebuah kartu yang dapat dibaca nomornya oleh sebuah program, dan programnya menentukan apakah nomornya ganjil atau genap agar siswa boleh masuk atau tidak, buatlah urutan langkah programnya.</p> <p>Deskripsi persoalan : program harus mengidentifikasi apakah sebuah bilangan (nomor siswa) ganjil atau genap.</p> <p>Sketsa Solusi : Bilangan genap adalah bilangan kelipatan 2. Oleh karena itu, bilangan genap atau ganjil dapat ditentukan dengan menghitung sisa pembagiannya dengan 2. Sisa pembagian suatu bilangan dengan 2 akan menghasilkan 0 atau 1. Jika sisanya 0 maka genap, sedangkan jika 1 maka ganjil</p> <p>Siswa bersama guru menuliskan hasil diskusi dari berbagai usulan siswa kemudian bersama2 menyimpulkan hasilnya yaitu sebagai berikut :</p> <p>Salah satu urutan langkah untuk mengidentifikasi apakah sebuah bilangan adalah bilangan ganjil atau bilangan genap adalah:</p> <ol style="list-style-type: none"> 1. Baca bilangan yang akan ditentukan, yaitu N sembarang bilangan. 2. M adalah hasil operasi Lakukan pengecekan dengan modulus 2 untuk mengetahui atau sisa hasil bagi dengan 2 terhadap N 3. Jika modulus/sisa hasil bagi bilangan dengan 2 yaitu M sama dengan 0 maka bilangannya adalah bilangan genap, jika M tidak sama dengan 0 maka bilangannya adalah ganjil 4. Tuliskan "Bilangan Genap" atau "Bilangan Ganjil" tergantung kepada hasil operasi modulo 2 (nilai M) pada langkah 3 <p>Guru memberi pertanyaan kembali kepada siswa : perhatikan langkah untuk menghitung bilangan bulat diatas, apabila langkah no 3 terlewatkan apakah bisa kita melakukan identifikasi bilangan ganjil dan genap? Dari diskusi tersebut diatas kita dapat menyimpulkan apa itu algoritma dan karakteristiknya.</p> <p>Siswa diminta menentukan cara lain untuk menentukan bilangan ganjil atau genap dengan menuliskan flowchart dan algoritmanya.</p>

KEGIATAN	DESKRIPSI KEGIATAN						
Inti	<p>Setelah memahami apa itu algoritma kemudian guru menjelaskan tentang cara penulisan algoritma untuk menentukan bilangan ganjil atau genap, yaitu sebagai berikut :</p> <ol style="list-style-type: none"> 1. Mulai 2. Deklarasikan variabel bernama N 3. Input nilai N 4. Lakukan pengecekan sisa pembagian N dengan 2 menggunakan modulus untuk mengetahui sisa hasil bagi dengan 2: <ol style="list-style-type: none"> 4.a. Jika sisa pembagian = 0 maka tuliskan “Bilangan adalah Genap” 4.b. Jika sisa pembagian=1 maka tuliskan “Bilangan adalah Ganjil” 5. Selesai <p>Kemudian dijelaskan kembali bagaimana flowchart untuk menentukan bilangan ganjil dan genap, yaitu sebagai berikut :</p>  <pre> graph TD Start([Mulai]) --> Input[/N/] Input --> Decision{N mod 2 = 0} Decision -- FALSE --> Output1[/Bilangan adalah Ganjil/] Decision -- TRUE --> Output2[/Bilangan adalah Genap/] Output1 --> End([Selesai]) Output2 --> End </pre> <p><u>Contoh lain Algoritma, notasi algoritma, flowchart dan Pseudocode.</u></p> <p>Apabila kita akan menghitung luas segi empat, maka akan didapatkan penyelesaian sebagai berikut :</p> <p><u>Notasi Algoritma :</u></p> <table border="1" data-bbox="419 1608 1393 2033"> <tr> <td>Judul</td> <td>/* menghitung program luas segi empat */</td> </tr> <tr> <td>Kamus</td> <td> luas : integer panjang : integer lebar : integer </td> </tr> <tr> <td>Algoritma</td> <td> Input : panjang Input : lebar Luas : panjang * lebar </td> </tr> </table>	Judul	/* menghitung program luas segi empat */	Kamus	luas : integer panjang : integer lebar : integer	Algoritma	Input : panjang Input : lebar Luas : panjang * lebar
Judul	/* menghitung program luas segi empat */						
Kamus	luas : integer panjang : integer lebar : integer						
Algoritma	Input : panjang Input : lebar Luas : panjang * lebar						

KEGIATAN	DESKRIPSI KEGIATAN				
	<p>Output : luas</p> <p>Dari notas algoritma diatas maka algoritmanya dari perhitungan luas segitiga adalah sebagai berikut:</p> <ol style="list-style-type: none"> 1. Mulai 2. Deklarasikan variabel panjang dan lebar 3. Masukkan nilai panjang dan nilai lebar 4. Hitung luas persegi yaitu $\text{Luas} = \text{panjang} * \text{lebar}$ 5. Nilai L (luas) akan dicetak sebagai output ke perangkat output (keluaran) <p>Kemudian hasil flowchart dan pseudocodenya adalah sebagai berikut :</p> <table border="1"> <thead> <tr> <th>FLOWCHART</th><th>PSEUDO CODE</th></tr> </thead> <tbody> <tr> <td>  <pre> graph TD Start([Mulai]) --> InputAlas[/Input alas/] InputAlas --> InputTinggi[/Input Tinggi/] InputTinggi --> Process[Luas = (alas*tinggi)/2] Process --> Output[Tampilkan Luas] Output --> End([Selesai]) </pre> </td><td> <pre> Algoritma LuasSetiga {Mencari_Luas_Segitiga} Deklarasi luas :integer panjang :integer lebar :integer Deskripsi read(panjang, lebar) luas = panjang * lebar write (luas) end </pre> </td></tr> </tbody> </table> <p>Algoritma ini akan menghitung luas kalau pengguna memasukkan nilai panjang dan lebar. Berikut ini adalah algoritma yang hanya menghitung luas kalau pengguna memasukkan nilai panjang dan tinggi postif.</p> <pre> Algoritma LuasSegitiga {Mencari_Luas_Segitiga} Deklarasi luas :integer panjang :integer lebar :integer Deskripsi read(panjang, lebar) if (panjang > 0 and lebar > 0) then luas = panjang * lebar write (luas) else write ("data salah, tidak dihitung") end end. </pre> <p>Setelah memahami bagaimana cara menulis sebuah algoritma, guru meminta siswa untuk berlatih untuk mengerjakan lembar kerja siswa dan diakhiri dengan pembahasan.</p>	FLOWCHART	PSEUDO CODE	 <pre> graph TD Start([Mulai]) --> InputAlas[/Input alas/] InputAlas --> InputTinggi[/Input Tinggi/] InputTinggi --> Process[Luas = (alas*tinggi)/2] Process --> Output[Tampilkan Luas] Output --> End([Selesai]) </pre>	<pre> Algoritma LuasSetiga {Mencari_Luas_Segitiga} Deklarasi luas :integer panjang :integer lebar :integer Deskripsi read(panjang, lebar) luas = panjang * lebar write (luas) end </pre>
FLOWCHART	PSEUDO CODE				
 <pre> graph TD Start([Mulai]) --> InputAlas[/Input alas/] InputAlas --> InputTinggi[/Input Tinggi/] InputTinggi --> Process[Luas = (alas*tinggi)/2] Process --> Output[Tampilkan Luas] Output --> End([Selesai]) </pre>	<pre> Algoritma LuasSetiga {Mencari_Luas_Segitiga} Deklarasi luas :integer panjang :integer lebar :integer Deskripsi read(panjang, lebar) luas = panjang * lebar write (luas) end </pre>				

KEGIATAN	DESKRIPSI KEGIATAN
Penutup	Guru dan siswa melakukan refleksi bersama dari materi yang telah disampaikan

7.1.5 Lembar Kerja Siswa

- Analisislah tentang bagaimana menghitung rugi laba pada sebuah toko bangunan. Data apa yang diperlukan untuk menghitung rugi laba ?
- Buatlah sebuah notasi algoritma tentang hal tersebut, gambarkan flowchart dan pseudocodenya

7.1.6 Asesment

Digunakan untuk menilai pada Lembar Kerja Siswa

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Membuat Notasi Algoritma	Dapat menyusun notasi algoritma secara runtut dan benar	Dapat menyusun notasi algoritma secara runtut dengan 1 kesalahan	Dapat menyusun notasi algoritma secara runtut dengan lebih dari 1 kesalahan	Tidak dapat menyusun notasi algoritma secara runtut dengan benar
Membuat flowchart	Dapat menyusun flowchart dari algoritma yang telah disusun dengan benar	Dapat menyusun flowchart dari algoritma yang telah disusun dengan 1 kesalahan	Dapat menyusun flowchart dari algoritma yang telah disusun dengan lebih dari 1 kesalahan	Tidak dapat menyusun flowchart dari algoritma yang telah disusun secara runtut dengan benar
Membuat pseudocode	Dapat menyusun pseudocode dari algoritma yang telah disusun	Dapat menyusun pseudocode dari algoritma yang telah disusun dengan 1 kesalahan	Dapat menyusun pseudocode dari algoritma yang telah disusun dengan lebih dari 1 kesalahan	Tidak dapat menyusun pseudocode dari algoritma yang telah disusun secara runtut dengan benar

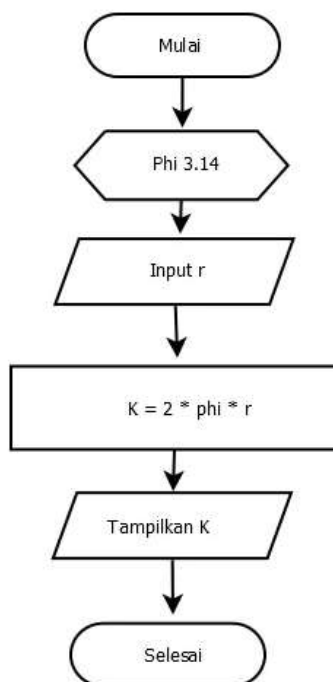
7.1.7 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa mampu menjelaskan tentang apa itu algoritma?	
Apakah siswa mampu membaca dan menulis dalam notasi algoritma ?	

Aspek	Refleksi Siswa
Apakah siswa mampu menyusun algoritma sebagai sketsa solusi persoalan?	
Apakah senang belajar algoritma?	
Apakah terdapat kesulitan dalam belajar algoritma?	

7.1.8 Contoh soal Ulangan

- a. Terdapat 2 buah gelas X dan Y yang berisi larutan berwarna . Gelas X berisi larutan berwarna ungu dan gelas Z berisi larutan berwarna merah. Volume di dua gelas tersebut sama banyaknya. Bagaimana cara menukarkan isi kedua gelas tersebut sehingga nantinya gelas X akan berisi larutan berwarna merah dan gelas Y berisi larutan berwarna ungu. Buatlah algoritmanya
- b. Perhatikan flowchart berikut :



Dari gambar flowchart diatas, proses perhitungan apa yang sedang diselesaikan?

- c. Dari soal B diatas buatlah notasi algoritma dan pseudocodenya.

7.2 Aktivitas-2 Ayo Mengenal IDE Pascal

Kode Aktivitas : K10-AP-P1-A2-IDEPascal

Pada aktivitas pembelajaran ini siswa akan di kenalkan dengan bahasa pemrograman pascal dan struktur penulisan program di dalamnya, menggunakan compiler online <https://ideone.com> dan kompiler **free pascal**. Setiap program kecil akan mewakili satu konsep pemrograman prosedural, dan merupakan program utuh yang siap dieksekusi. Keseluruhan aktivitas diperkirakan akan memerlukan 1 minggu.

Aktivitas ini akan ditutup dengan latihan pembuatan program sapaan menggunakan free pascal dan atau compiler onlien **ideone.com**.

7.2.1 Pertanyaan Pemantik

Apakah kalian pernah memakai ponsel? Ada aplikasi apa saja pada ponsel yang pernah kalian pakai? Tahukah bahwa aplikasi pada ponsel adalah sebuah program komputer yang dibangun menggunakan bahasa pemrograman, kemudian programnya dipindahkan ke ponsel ?

7.2.2 Konsep

Setelah belajar memprogram visual di SMP, kalian sekarang akan belajar pemrograman prosedural tekstual. Pada pemrograman tekstual, kalian tidak mengkomposisi blok siap pakai, tetapi harus mengetikkan perintah dalam bentuk teks, yang kosa katanya biasanya adalah dalam bahasa Inggris. Kalian akan belajar salah satu bahasa pemrograman tekstual, yang disebut bahasa Pascal.

SEJARAH BAHASA PASCAL

Pascal adalah bahasa pemrograman imperatif dan prosedural, yang dirancang oleh Niklaus Wirth sebagai bahasa sederhana, “kecil” dan efisien yang dimaksudkan untuk mendorong praktik pemrograman yang baik menggunakan pemrograman terstruktur dan pengorganisasian data. Untuk menghormati ahli matematika, filsuf, dan fisikawan Prancis yang bernama Blaise Pascal, bahasa ciptaan Prof. Niklaus Wirth ini dinamakan Bahasa Pascal.

MENGAPA BAHASA PASCAL YANG DIPAKAI

Kalian tentunya penasaran, kenapa bahasa Pascal yang akan dipelajari ?

Alasan utama kenapa bahasa Pascal dipilih adalah karena Bahasa Pascal adalah bahasa pemrograman kecil dan sederhana yang mudah dipelajari bagi pemula, dan sebuah bahasa yang ketat type.

Bahasa Pascal adalah bahasa ketat type (pengguna harus menyatakan type secara eksplisit setiap kali mendeklarasikan variabel atau konstanta) akan memberikan “keamanan” proses komputasi. Kelak, kalau kalian sudah mengalami memprogram dalam berbagai bahasa, kalian akan merasakan bahwa bahasa-bahasa yang ketat type (*“strong type” programming language*) akan memberikan keamanan ketimbang bahasa yang “bebas type” (*weak type programming language*) yang mungkin lebih “nyaman” dituliskan.

Selain ketat type, salah satu hal yang “menyebalkan” dalam menulis program dalam bahasa pascal adalah bahwa pengguna harus menuliskan “begin” dan “end” yang menentukan scope dengan jelas. Tapi zaman sekarang, begin dan end banyak dibantu pengetikannya oleh IDE yang digunakan. Coba, kalian masuk ke lingkungan <https://ideone.com/> dan memilih bahasa Pascal, kerangka program pascal siap kalian isi. Mudah bukan ?

Walaupun bahasa pascal sangat jarang dipakai untuk memprogram aplikasi, bahasa pascal sangat baik dipakai untuk edukasi terutama bagi pemula. Bahasa pemrograman adalah bahasa yang sederhana dengan aturan sangat baku, tidak serumit bahasa natural. Setelah menguasai salah satu bahasa pemrograman, kita akan dapat berpindah ke bahasa lain yang sejenis dengan mudah.

ATURAN DALAM PENULISAN PROGRAM PASCAL

A. Perbedaan Huruf Besar / Kecil

Pascal tidak membedakan penggunaan huruf besar atau kecil. Dalam pemrograman, hal ini dikenal dengan istilah case insensitive. Kita boleh menulis `program`, `Program`, `PROGRAM`, `WRITEln`, maupun `writeln`. Semuanya dianggap sama dalam bahasa pascal. Ini memudahkan bagi pemula.

B. Cara Penulisan Komentar di dalam Pascal

Komentar atau comment adalah 'kode program' yang ditambahkan untuk memberi keterangan/penjelasan mengenai cara kerja program. Komentar tidak akan diproses oleh Pascal dan berfungsi hanya untuk memberi keterangan tambahan, terutama jika kode program yang ditulis cukup rumit.

Untuk membuat komentar di dalam kode program pascal, terdapat 2 alternatif pilihan:

- Kurung dan bintang, komentar boleh lebih dari 1 baris: `(* di sini komentar *)`
- Garis miring ganda untuk komentar yang muat dalam 1 baris: `// di sini komentarnya`
- Kurung kurawal komentarnya boleh lebih dari 1 baris: `{ ini komentar }`

Contoh :

```
Program hello_word;
begin
    (*kode untuk menampilkan tulisan 'Hello Word'*)
    writeln('Hello Word');
end.
```

C. Penggunaan Whitespace

Whitespace adalah istilah pemrograman yang merujuk kepada tanda baca 'spasi' yang tidak terlihat. Contoh karakter-karakter *whitespace* adalah: spasi, tab dan enter (new line). Di dalam Pascal, secara umum *whitespace* akan diabaikan.

D. Statement di dalam Pascal

Statement adalah sebuah baris perintah yang bisa melakukan sebuah tindakan, apakah itu menampilkan teks di layar, meminta input, perulangan, percabangan program (logika IF), dll. Sebuah statement di dalam Pascal harus diakhiri dengan tanda titik koma ";" (kecuali untuk beberapa kondisi khusus).

E. Expression di dalam Pascal

Expression adalah potongan kode program yang menghasilkan suatu nilai. *Expression* pada dasarnya merupakan bagian dari sebuah statement.

Contoh *arithmetic expression* : `4+9`
 `8/4`

Contoh *boolean expression* : `found and OK`
 `true or false`

F. Identifier di dalam Pascal

Identifier adalah "nama", bagian dari statement yang merupakan 'identitas' dari sesuatu. Identitas ini meliputi:

- Nama program

- b. Nama konstanta
- c. Nama variabel
- d. Nama fungsi atau prosedur

Aturan penulisan *identifier* adalah sebagai berikut:

- a. Karakter pertama harus berupa huruf.
- b. Karakter kedua dan seterusnya bisa berupa huruf, angka, atau karakter *underscore* “_”.
- c. Maksimal panjang *identifier* tergantung kepada compiler yang digunakan. Beberapa mendukung 32 karakter, namun kebanyakan mendukung hingga 63 karakter. Jika anda membuat *identifier* dengan panjang melebihi 63 karakter, hanya 63 karakter pertama saja yang akan digunakan.
- d. Penulisan *identifier* tidak boleh menggunakan karakter selain angka, huruf dan *underscore*. Kita tidak bisa menggunakan spasi, dan tanda-tanda khusus seperti *, +, -, &, ^, %, \$, #, atau @.

Berikut adalah contoh penulisan *identifier*:

Penulisan <i>identifier</i> yang benar	Penulisan <i>identifier</i> yang salah dan penjelasannya	
nama_pengguna luas_segitiga PanjangLingkaran angKatan45 NAMAKOTA	6siswa 5+7 Pertama* Luas segitiga	Diawali angka Mengandung karakter ‘+’ Mengandung karakter ‘*’ Mengandung spasi

G. Reserved Word dan Predefined Identifier

Reserved Word dan *Predefined Identifier* adalah kata/karakter khusus yang digunakan secara internal di dalam aplikasi Pascal. Kita tidak boleh menggunakan kata-kata ini sebagai *identifier*.

Berikut adalah *Reserved Word* di dalam Pascal:

absolute	and	array	asm	begin	break	case	const
constructor	continue	destructor	div	do	downto	else	end
file	for	function	goto	if	implementation	in	inherited
inline	interface	label	mod	nil	not	object	of
on	operator	or	packed	procedure	program	record	reintroduce
repeat	self	set	shl	shr	string	then	to
type	unit	until	uses	var	while	with	xor

Reserved word adalah kata kunci yang dikenal dalam pascal, yang mempunyai arti tertentu. Perhatikan bahwa dengan beberapa kata kunci itu, kita dapat membuat program apapun. Bandingkan dengan kosa kata bahasa Indonesia atau bahasa Inggris yang jumlahnya banyak sekali (puluhan ribu)!

STRUKTUR DASAR PROGRAM PASCAL

Bahasa Pascal adalah bahasa pemrograman deklaratif dan terstruktur, yang artinya seluruh variabel, konstanta, fungsi, dan beberapa struktur pemrograman lain harus ‘diperkenalkan’, dinyatakan secara eksplisit diawal kode program dan dalam urutan yang sudah ditentukan. Didalam pemrograman, ‘perkenalan’ atau ‘pernyataan eksplisit’ ini dikenal dengan istilah ‘deklarasi’ (*declarations*).

Struktur kode program pascal bisa dipecah menjadi 2 bagian: Kepala Program dan Tubuh Program.

```
program nama_program;  
    { bagian deklarasi }  
  
begin  
    { main program }  
end.
```

A. KEPALA PROGRAM

“Kepala” (*header*) program terdiri dari judul program dan deklarasi.

A.1. JUDUL PROGRAM

Judul program Pascal bersifat opsional. Program masih bisa berjalan, meskipun belum diberikan judul. Namun demikian, sebaiknya nama program dituliskan agar kita dapat mengenali program dengan mudah, seperti menuliskan judul dalam teks.

Tujuan dari penulisan judul program ini adalah agar pembaca program mengetahui program “apa” yang dilakukan oleh program, yang “bagaimana” atau langkah rincinya ditulis pada badan program.

Nama program ditulis pada deret paling atas. Sebaiknya nama program singkat dan jelas. Namun demikian, apabila terdiri dari beberapa suku kata, maka dipisah dengan garis bawah (). Contoh : Program hitung_Lingkaran

A.2. DEKLARASI

Bagian deklarasi terdiri dari beberapa jenis program diantaranya

a. Perintah `uses`

Uses adalah cara bahasa pemrograman pascal untuk memasukkan kode-kode eksternal yang dikenal dengan unit (atau *library* dalam bahasa pemrograman lain). Sebagai contoh, perintah `clrscr` yang digunakan pada program utama merupakan perintah yang ada pada unit `crt`. Terdapat berbagai unit yang bisa kita gunakan, seperti `math`, `sysutils`, `printer`, dan `strutils`.

Contoh : `Uses crt;`

b. Deklarasi Tipe (*Type*)

Pada bagian deklarasi `Type`, kita bisa membuat tipe data bentukan. Contoh: tipe data warna, dimana tipe data ini hanya bisa diisi dengan nilai merah, kuning, atau biru. Deklarasi `Type` belum dibahas pada modul pemrograman dasar ini. Kita hanya memprogram dengan menggunakan tipe yang sudah disediakan dalam bahasa Pascal yaitu: `integer`, `real`, `char`, dan `string`.

c. Deklarasi Konstanta (*constant*)

Konstanta atau *constant* adalah sebuah nama yang 'menampung' suatu nilai yang bersifat tetap. Contoh : PI dengan nilai 3.14.

d. Deklarasi Variabel (*variable*)

Variabel atau *variable* adalah sebuah nama yang 'menampung' nilai, dan nilai yang ditampung tersebut bisa diubah.

e. Deklarasi Fungsi (*function*)

Sebuah fungsi atau *function* dikenal juga sebagai subprogram, yaitu pengemasan kode program yang panjang menjadi fungsi-fungsi yang saling terpisah dan dapat dipanggil untuk melakukan komputasi. Berbeda dengan prosedur, fungsi akan mengembalikan nilai.

f. Deklarasi Prosedur (*procedure*)

Pada dasarnya, prosedur atau *procedure* adalah sederetan instruksi yang diberi nama, untuk dapat dipanggil oleh program utama atau prosedur/fungsi lain agar program dapat didekomposisi menjadi bagian-bagian bermakna, sehingga dalam program pemanggil, kode programnya lebih mudah dibaca dan dipahami.

B. TUBUH PROGRAM

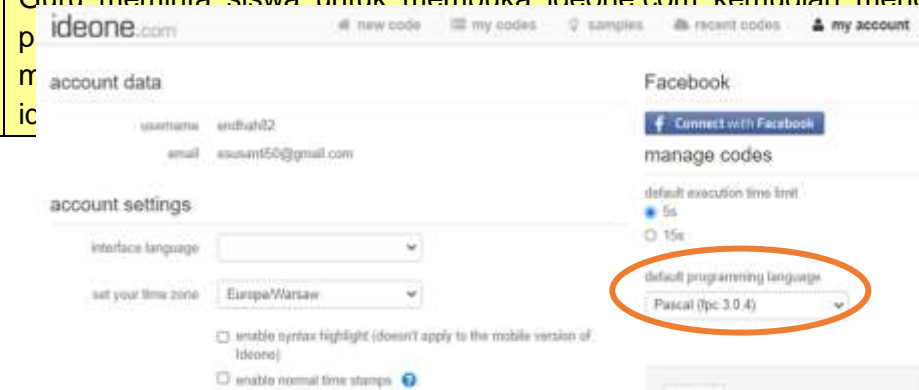
Tubuh (*body*) program terdiri dari program utama (*main program*) yang berisi kode program utama dimana kode program utama ditulis. Disinilah dilakukan deklarasi konstanta, variabel penulisan serta pemanggilan fungsi atau prosedur (jika ada). *Main program* diawali dengan *keyword* `begin` dan diakhiri dengan `end`.


7.2.3 Kata Kunci


Program, program utama (*main program*), *source code*, bahasa pemrograman, bahasa pemrograman Pascal, bahasa pemrograman ketat tipe (*strong type*)

7.2.4 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN
Pendahuluan	Pada awal pembelajaran, guru memberikan materi tentang IDE pascal kemudian bediskusikan dan menjelaskan tentang materi yang sudah dibaca siswa
Inti	Guru meminta siswa untuk membuka ideone.com kemudian mengatur



KEGIATAN	DISKRIPSI KEGIATAN
	<p>yang sudah siap akan mengalami proses kompilasi, yaitu diperiksa apakah salah sintaks, kemudian dieksekusi (<i>run</i>).</p> <ul style="list-style-type: none"> ▪ Salah sintaks adalah penulisan program yang tidak sesuai dengan aturan bahasa Pascal, misalnya :identifier yang tidak sesuai aturan. ▪ Begin yang tidak berpasangan dengan end ▪ Kurung yang tidak berpasangan ▪ Pemakaian nama yang belum dideklarasasi ▪ Penulisan instruksi tidak sesuai aturan <p>Program yang salah sintaks belum tentu benar. Contohnya saat menghitung luas segi4, tetapi kita menuliskan :</p> <p>Luas := panjang + lebar;</p> <p>Walaupun program lolos salah sintaks, program akan menghitung luas segi empat dengan rumus yang salah karena rumus/ekspresi untuk menghitung luas salah, seharusnya :</p> <p>Luas := panjang * lebar;</p> <p>Kesalahan ini disebut kesalahan semantik.</p> <p>Kompiler hanya dapat memeriksa kesalahan sintaks. Oleh sebab itu, kita harus teliti dalam menulis program. Kesalahan ketik walaupun hanya 1 karakter saja, bisa berakibat fatal.</p> <p>Guru meminta siswa untuk mengetikkan code Program P-AP-01 'hello' berikut kemudian mencompilanya</p> <pre> program hello; (* File : HELLO.PAS *) (* menuliskan Hello ke layar *) Begin writeln ('hello ') ; end. </pre> <p>Berikut hasil tampilan pada free pascal :</p>  <p>Berikut tampilan pada ideone.com</p>

KEGIATAN	DISKRIPSI KEGIATAN										
	<div data-bbox="422 230 1034 674">  <pre> 1. program hello; 2. (* File : HELLO.PAS *) 3. (* menuliskan Hello ke layar *) 4. begin 5. writeln ('hello '); 6. end. 7. </pre> <p>Success #stdin #stdout 0s 5460KB</p> <p>stdin hello</p> <p>stdout hello</p> </div> <p>Setelah selesai mengetikkan program dan meng-kompilasinya, siswa melihat apakah program bisa berjalan dengan baik atau tidak. Apabila terjadi kesalahan sintaks maka siswa harus belajar melacak statement yang mana yang membuat program tidak bisa berjalan.</p> <p>Selanjutnya setelah siswa dapat menjalankan programnya dengan output yang diharapkan sudah tertampil maka guru berdiskusi dengan siswa tentang fungsi-fungsi dari setiap baris kode program yang sudah dibuat dan di <i>compile</i> dengan sukses. Siswa dapat mengetahui kebenaran program dari tampilan dari kode program yang sudah di buat.</p> <p>Guru menyimpulkan diskusi sebagai berikut :</p> <p>Sebuah kode program pascal, diawali dengan keyword program kemudian diikuti dengan judul program yang diinginkan (dalam contoh diatas, saya menggunakan judul hello). Selanjutnya, kode program dibuka dengan perintah “begin”, dan diakhiri dengan perintah “end.” Diantara “begin” dan “end.” inilah seluruh kode program pascal dituliskan.</p> <p>Contoh program yang diatas dapat diterjemakan setiap perintahnya sebagai berikut :</p> <table border="1" data-bbox="422 1496 1390 1738"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td>Program hello;</td><td>Judul program dengan nama hello</td></tr> <tr> <td>Begin</td><td>Tanda awal program utama</td></tr> <tr> <td>writeln ('hello ') ;</td><td>Perintah untuk menampilkan teks 'hello'</td></tr> <tr> <td>End.</td><td>Mengakhiri sebuah program, selalu diakhiri dengan tanda “.” (titik)</td></tr> </tbody> </table> <p>Jika menggunakan IDE, setelah menulis program akan keluar,.... Fungsi readln digunakan agar hasil yang dimunculkan dapat dibaca, karena jika tidak maka hasil akan muncul kurang dari 1 detik (tidak bisa terlihat).</p>	KODE PROGRAM	MAKNA	Program hello;	Judul program dengan nama hello	Begin	Tanda awal program utama	writeln ('hello ') ;	Perintah untuk menampilkan teks 'hello'	End.	Mengakhiri sebuah program, selalu diakhiri dengan tanda “.” (titik)
KODE PROGRAM	MAKNA										
Program hello;	Judul program dengan nama hello										
Begin	Tanda awal program utama										
writeln ('hello ') ;	Perintah untuk menampilkan teks 'hello'										
End.	Mengakhiri sebuah program, selalu diakhiri dengan tanda “.” (titik)										

KEGIATAN	DISKRIPSI KEGIATAN
	<p>Setelah selesai dengan kegiatan pertama , selanjutnya guru meminta siswa untuk berlatih mengetikkan kode program yang sedikit berbeda. Kode Program P-AP-02 yang dibuat adalah sebagai berikut :</p> <pre> program hellodos; (* File : HELLODOS.PAS *) (* menuliskan Hello ke layar *) uses crt; begin clrscr; writeln ('hello ') ; end. </pre> <p>Dari program diatas terdapat beberapa perintah yang harus kita pahami yang tidak terdapat dalam program hello yang sebelumnya yaitu perintah : <code>Uses crt, Clrscr.</code></p> <p>Perintah <code>clrscr</code> ini berasal dari unit <code>crt</code>. <code>Clrscr</code> merupakan singkatan dari <i>clear screen</i>, digunakan untuk menghapus output dari kode program sebelumnya (membersihkan layar).</p> <p>Setelah siswa selesai mengerjakan 2 latihan diatas guru meminta siswa untuk membandingkan kedua latihan tersebut. Siswa diharapkan dapat menyimpulkan perbedaan keduanya source code tersebut bahwa :</p> <ol style="list-style-type: none"> Untuk latihan pertama, menuliskan hello ke layar (Pascal standar) Untuk latihan kedua, menuliskan hello ke layar (Lingkungan DOS)
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisi pada form yang sudah dibuat

7.2.5 Lembar Kerja Siswa

Tujuan :

siswa dapat membuat kode program untuk pengenalan diri seperti hasil yang sudah ditentukan pada soal melalui ideone.com

Petunjuk Pengerjaan :

- Buatlah program pascal untuk dapat menampilkan output program seperti berikut ini ganti "ENDAH SUSANTI" dengan namamu:

```

Namaku : ENDAH SUSANTI
Kelasku : X MIPA 1
=====
AKU SUKA BELAJAR PASCAL

```

Serahkan source code yang dibuat dan Screenshoot hasil output program ,

7.2.6 Asesment

Rubrik ini digunakan untuk menilai pada Lembar Kerja Siswa

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Mampu mengetik program dengan benar	Siswa mampu mengetikkan 4 baris program pengenalan seperti tampilan	Siswa mampu mengetikkan 3 baris program pengenalan seperti tampilan	Siswa mampu mengetikkan 2 baris program pengenalan seperti tampilan	Siswa mampu mengetikkan 1 baris program pengenalan seperti tampilan
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sesekali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

7.2.7 Pengayaan

Sebagai pengayaan pengetahuan tentang IDE Pascal lakukan latihan mandiri seperti petunjuk dibawah ini :

- Instal program freepascal pada komputer masing-masing. Aplikasi freepascal dapat di download melalui <https://www.freepascal.org/download.html>
- Buatlah program pascal yang berisi sapaan dengan output seperti dibawah ini kemudian *compile* dan *Run* menggunakan free pascal.

```
Namaku : ENDAH SUSANTI
Kelasku : X MIPA 1
=====
AKU SUKA BELAJAR PASCAL
```

- Pastikan program dapat berjalan dengan baik, serahkan file *source code* sesuai yang ditentukan oleh guru.

7.2.8 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami struktur program utama pada bahasa pascal?	
Apakah siswa mampu membuat kode program "Hello"?	

Jika menemui kesalahan, apakah siswa mampu secara mandiri mendeteksi kesalahan program yang tidak bisa berjalan dengan baik?	
Apakah senang belajar bahasa pemrograman pascal?	
Apakah terdapat kesulitan dalam belajar bahasa pemrograman pascal?	

7.2.9 Contoh Soal Ulangan

Cermatilah kode program (*source code*) berikut :

```
Program AkuBahagia
Uses crt;
Begin;
    Clrscr;
    write('Aku Bahagia');
End.
```

Pertanyaan:

- Jelaskan makna dari setiap baris *source code* yang ada di dalamnya
- Tampilan output yang akan muncul dari *source code* di atas adalah
- Jika dituliskan ke dalam IDE ideone.com, apakah kode tersebut dapat menampilkan hasil dengan benar dan tidak menimbulkan *error*?
Jika soal (c) di jawab *error*, jelaskan kesalahan apa yang terdapat dalam *source code* tersebut.

7.3 Aktivitas-3 Program Input Output

Kode Aktivitas : K10-AP-P2-A3-InputOutput

Pada aktivitas ini, siswa akan belajar tentang program input- output serta tipe data dasar pada sebuah program. Guru menampilkan program kecil untuk proses input-output yang dapat membantu siswa dalam memahami makna kode program . Keseluruhan aktivitas diperkirakan akan memerlukan 1 pertemuan.

Materi yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pembuatan program menampilkan data dengan berbagai jenis tipe data menggunakan free pascal atau ideone.com.

7.3.1 Pertanyaan Pemantik

Apakah kalian pernah mengambil uang di ATM? Kalian pasti akan memasukkan sebuah kartu, kemudian melakukan apa yang diminta oleh mesin ATM dan akhirnya mendapatkan keluaran yang kalian inginkan yaitu uang. Tahukah bahwa “di dalam” mesin ATM ada program komputernya?

7.3.2 Kata Kunci

Input, output, write, Writeln, read, readln

7.3.3 Konsep Terkait Aktivitas

Sebagian besar program aplikasi membutuhkan *input* dan menghasilkan *output*. Suatu program yang tidak dapat menghasilkan bentuk keluaran (*output*) akan sulit dites kebenarannya, karena hasil dari proses yang dilakukan oleh program tersebut tidak dapat kita lihat. Walaupun program tidak menghasilkan output, biasanya kita menuliskan “sesuatu” untuk mentest.

Namun tidak semua program harus mengandung *input-output*, sebagai contoh adalah *Operating system* dia tidak dapat dilihat prosesnya.

Untuk program input-output ini kita akan belajar tentang perintah : **write**, **writeln**, **read**, **readln**.

A. WRITE DAN WRITELN

A.1. FUNGSI & PERBEDAAN WRITE DAN WRITELN

Write dan **writeln** sama-sama digunakan untuk menampilkan ‘sesuatu’ dari dalam kode pascal ke jendela tampilan, atau dalam istilah pemrograman digunakan sebagai perintah ‘ouput’. Perbedaan antara write dan writeln terletak pada apakah ‘output’ selanjutnya ditampilkan pada baris yang sama, atau di baris baru.

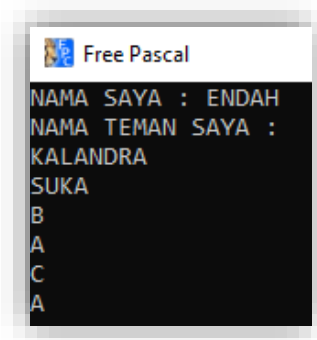
Perintah **write** akan menampilkan ‘output’, dan kursor teks tetap berada di baris yang sama sehingga penulisan berikutnya akan dilanjutkan pada baris yang sama).

Sedangkan perintah **writeln** akan menampilkan ‘output’, kemudian melakukan “ganti baris” (kursor teks akan pindah ke baris berikutnya sehingga penulisan berikutnya dilakukan pada baris berikutnya (ganti baris).

Data-ouput ini bisa berupa teks, variabel, konstanta, dll. Agar lebih mudah dipahami, berikut contoh kode Program P-AP-03 sebagai berikut yang memperlihatkan perbedaan kedua perintah ini:

```
program tampil;
(* File :tampil.PAS *)
(* Program P-AP-03: contoh tampilan *)
uses crt;
begin
  clrscr;
  write('NAMA SAYA : ');
  writeln('ENDAH');
  writeln('NAMA TEMAN SAYA : ');
  writeln('KALANDRA');
  write('S');
  write('U');
  write('K');
  writeln('A');
  writeln('B');
  writeln('A');
  writeln('C');
  writeln('A');
end.
```

Hasil output yang akan muncul adalah :



Dari hasil tampilan output diatas dapat disimpulkan beberapa hal yaitu :

Dapat dilihat hasil kode program pada baris 1 dan 2 ditampilkan dalam 1 baris (pada hasil NAMA SAYA : ENDAH) Ini karena perintah **write** akan membuat teks berikutnya tetap di baris yang sama.

Hasil tampilan juga memperlihatkan kata "BACA", dapat tertampil satu baris 1 huruf. Ini terjadi karena perintah **writeln** hanya akan memindahkan teks selanjutnya ke baris baru; **writeln** artinya **write** kemudian ganti baris (**newline**).

A.2. CARA PENULISAN WRITE DAN WRITELN

Data-output yang ingin ditampilkan dengan perintah **write** dan **writeln**, harus ditulis diantara tanda kurung " (" dan ") ".

Jika data tersebut adalah 'teks' yang terdiri dari karakter (**char**) atau kumpulan karakter (**string**) kita perlu menambahkan tanda kutip satu (**`**) diantara teks tersebut. Apabila yang akan ditampilkan angka, variabel, atau konstanta, kita tidak perlu menggunakan tanda kutip, artinya yang akan ditampilkan adalah isinya. Berikut contoh kode programnya adalah P-AP-04 sebagai berikut:

```
program cara_tulis_write;
(* File :tuliswrite.PAS *)
(* Program P-AP-04: contoh masukan *)
uses crt;
const
    kota='Yogyakarta';
var
    nama:string='Santi';
    umur:integer=19;
begin
    clrscr;
    write('Nama : ');
    writeln(nama);
    write('Umur : ');
    writeln(umur);
    write('Kota : ');
    writeln(kota);
    readln;
end.
```

Perintah `write('namaku');` berbeda dengan `write(namaku);`

. Penulisan pertama berarti kita ingin menampilkan teks 'namaku', sedangkan penulisan yang kedua kita ingin menampilkan nilai dari variabel/konstanta yang bernama 'namaku'.

B. READ DAN READLN

B.1. FUNGSI DAN PERBEDAAN READ DAN READLN

Read dan **readln** berfungsi untuk memasukkan 'sesuatu' ke dalam kode program. Di dalam pemrograman, ini disebut sebagai perintah 'input'.

Perintah **read** akan membaca data secara 'horisontal'. Setelah proses input selesai, posisi kursor akan tetap berada di baris yang sama. Kita bisa menggunakan karakter 'spasi' atau 'enter' untuk memisahkan 1 input dengan input lainnya (dengan beberapa pengecualian).

Perintah **readln** akan membaca data secara 'vertikal'. Setelah satu data input selesai diketikkan, pengguna harus menekan 'enter' karena posisi cursor akan pindah ke baris baru. Kita bisa menggunakan karakter 'enter' untuk memisahkan satu input dengan input lainnya.

B.2. CARA PENULISAN READ DAN READLN

Untuk dapat menggunakan perintah **read** dan **readln**, kita harus mempersiapkan variabel yang akan menampung hasil input dari pengguna. Variabel ini juga harus memiliki tipe data yang sama dengan apa yang akan diinput.

Sebagai contoh, jika kita mengharapkan pengguna untuk memasukkan nama, maka variabel penampung harus bertipe `string`, namun jika kita meminta input umur, maka harus menggunakan variabel bertipe `integer`.

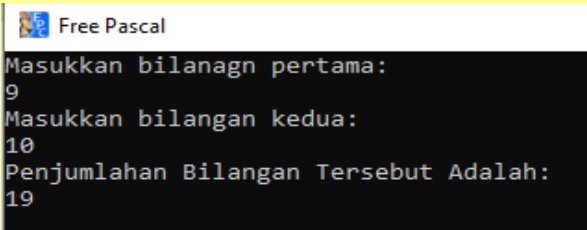
Agar lebih memahami konsep ini, berikut adalah contoh kode Program P-AP-05 penggunaan kedua perintah ini:

```
program baca;
(* File : BACA.PAS *)
(* contoh membaca integer*)
(* kemudian menuliskan nilai yang dibaca *)
uses crt;
var
    a : integer;
begin
    clrscr;
    (* Program *)
    writeln ('Contoh membaca dan menulis, ketik nilai
integer:');
    readln (a) ;
    writeln ('nilai yang dibaca : ', a) ;
end.
```

7.3.4 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN														
Pendahuluan	Pada awal pembelajaran, guru memberikan materi tentang apa itu input-output dalam bahasa pemrograman. Kemudian guru meminta siswa untuk membuka ideone.com untuk bisa mencoba mempraktikkan kode program input-output.														
Inti	<p>Bacalah dengan cermat contoh Program P-AP-06 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre> program baca; (* program P-AP-06:contoh membaca data *) (* File : BACA1.PAS *) (* contoh membaca integer*) (* kemudian menuliskan *) (* nilai yang dibaca *) (* Kamus *) var v1, v2,v3: integer; begin (* Program *) writeln ('Contoh membaca dan menulis, ketik nilai integer:'); readln (v1) ; writeln ('nilai yang dibaca : ', v1) ; readln (v2); writeln ('nilai yang dibaca : ', v3) ; readln (v3); end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada disana, dengan kesimpulan sebagai berikut :</p> <table> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> <tr> <td>Program baca;</td><td>Merupakan judul program dengan nama baca</td></tr> <tr> <td>Begin</td><td>Tanda awal badan program</td></tr> <tr> <td>Var a : integer</td><td>Deklarasi variabel a dengan tipe data integer (angka bulat)</td></tr> <tr> <td>writeln ('Contoh membaca dan menulis, ketik nilai integer:');</td><td>Perintah untuk menampilkan output berupa teks 'Contoh membaca dan menulis, ketik nilai integer'</td></tr> <tr> <td>readln (a) ;</td><td>Program meminta input sebuah bilangan bulat, disimpan dalam variabel bernama a</td></tr> <tr> <td>writeln ('nilai yang dibaca : ', a) ;</td><td>Perintah untuk menampilkan teks 'nilai yang dibaca :</td></tr> </table>	KODE PROGRAM	MAKNA	Program baca;	Merupakan judul program dengan nama baca	Begin	Tanda awal badan program	Var a : integer	Deklarasi variabel a dengan tipe data integer (angka bulat)	writeln ('Contoh membaca dan menulis, ketik nilai integer:');	Perintah untuk menampilkan output berupa teks 'Contoh membaca dan menulis, ketik nilai integer'	readln (a) ;	Program meminta input sebuah bilangan bulat, disimpan dalam variabel bernama a	writeln ('nilai yang dibaca : ', a) ;	Perintah untuk menampilkan teks 'nilai yang dibaca :
KODE PROGRAM	MAKNA														
Program baca;	Merupakan judul program dengan nama baca														
Begin	Tanda awal badan program														
Var a : integer	Deklarasi variabel a dengan tipe data integer (angka bulat)														
writeln ('Contoh membaca dan menulis, ketik nilai integer:');	Perintah untuk menampilkan output berupa teks 'Contoh membaca dan menulis, ketik nilai integer'														
readln (a) ;	Program meminta input sebuah bilangan bulat, disimpan dalam variabel bernama a														
writeln ('nilai yang dibaca : ', a) ;	Perintah untuk menampilkan teks 'nilai yang dibaca :														

KEGIATAN	DISKRIPSI KEGIATAN	
		dan diikuti nilai yang tersimpan dalam variabel a
	End.	Akhir program utama, selalu diakhiri dengan tanda titik ('.')
	<p>Pertanyaan :</p> <p>a. Kenapa sebelum membaca sebuah nilai, sebaiknya diawali dengan penulisan pesan ?</p> <p>b. Kenapa sebaiknya hasil membaca di outputkan ?</p> <p>Setelah selesai memahami tentang latihan input-output selanjutnya guru memberikan materi tentang tipe data yang sering digunakan dalam bahan pemrograman pascal.</p> <p>Bacalah dengan cermat contoh Program P-AP-07 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre>Program Penjumlahan_Angka; (*File :PENJUMALAHN.PAS*) (*contoh menghitung integer*) (*kemudian menuliskan nilai yang dibaca*) Uses crt; var Bill: integer; Bil2: integer; Jumlah: integer; Begin Clrscr; (*Bagian Input*) write('Masukkan Bilangan Pertama:'); readln(Bil1); write('Masukkan Bilangan Kedua:'); readln(Bil2); (*Bagian Proses*) Jumlah:=Bil1+Bil2; (*Bagian Output*) writeln('Jumlah Bilangan =',Jumlah); end.</pre> <p>Kemudian guru bersama siswa mendiskusikan perintah-perintah yang ada pada kode program P-AP-07, dengan kesimpulan sebagai berikut :</p>	
	KODE PROGRAM	MAKNA
	Program Penjumlahan_Angka;	Nama program : Penjumlahan_Angka
	Begin	Awal program utama

KEGIATAN	DISKRIPSI KEGIATAN	
	<pre>Var Bil1: integer; Bil2: integer; Jumlah: integer;</pre>	Deklarasi tiga buah variabel Bil1, Bil2, Jumlah bertipe integer (bilangan bulat)
	<pre>write('Masukkan Bilangan Pertama:');</pre>	Perintah untuk menampilkan teks 'Masukkan bilangan pertama'
	<pre>readln(Bil1);</pre>	Program meminta input untuk disimpan dalam variabel Bil1.
	<pre>write('Masukkan Bilangan Kedua:');</pre>	Perintah untuk menampilkan teks 'Masukkan bilangan kedua'
	<pre>readln(Bil2);</pre>	Program meminta input untuk disimpan dalam variabel Bil2.
	<pre>Jumlah:=Bil1+Bil2;</pre>	Penjumlahan Bil1+Bil2 (ekspresi aritmatika penjumlahan), dan menyimpan hasil penjumlahan pada variabel Jumlah dengan ekspresi ':=' (assignment)
	<pre>writeln('Jumlah Bilangan =', Jumlah);</pre>	Perintah untuk menampilkan teks 'Jumlah bilangan =' dan diikuti hasil yang tersimpan di variabel Jumlah yang sudah diproses dengan menjumlahkan
	<pre>End.</pre>	Akhir program utama (selalu diakhiri dengan tanda titik).
<p>Aktivitas Kreatif Siswa:</p> <p>a. Latihan pemahaman nama variabel Pada Kode Program XXX : Gantilah teks Bil1, Bil2, Jumlah; menjadi nama variabel yang lain; bagian teks mana sajakah yang harus diganti agar program tetap bisa berjalan.</p> <p>b. Latihan pemahaman perintah <code>writeln</code>, <code>readln</code>: Ubahlah program sehingga angka 19 dapat tampil di bawah tulisan penjumlahan bilangan tersebut, seperti gambar output dibawah ini. Bagian program yang mana yang harus kita ubah?:</p>  <p>c. Latihan pemahaman tipe data</p>		

KEGIATAN	DISKRIPSI KEGIATAN
	Apabila deklarasi variabel <code>Bil1</code> , <code>Bil2</code> , <code>Jumlah</code> diganti dengan type data <code>boolean</code> , apakah program bisa berjalan dengan baik? Berikan alasannya!
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisikan pada form yang sudah dibuat

7.3.5 Lembar Kerja Siswa

(1) Latihan *Problem Solving* 1 – Kebun Pak Arman

Pak Arman sedang memperhatikan ladangnya yang berbentuk persegi, agar bisa digunakan untuk membuat usaha kebun pisang, diperkirakan ladang tersebut dapat menampung 100 pohon pisang yang setiap panennya bisa memperoleh keuntungan semaksimal mungkin. Buatlah identifikasi input, proses dan output untuk menghitung keliling persegi, kemudian tulis program untuk menghitung luas kebun pak arman yang berbentuk persegi panjang tersebut dengan satuan angka yang digunakan adalah cm.

(2) Latihan *Problem Solving* 2 – Data Kartuku

Amatilah sebuah Kartu Identitas sebuah KTP, SIM atau kartu lain. Buatlah program pascal yang dapat menampilkan output sesuai dengan data yang ada di sana. (Tanpa menampilkan foto/gambar. Identifikasi input dan output, sebelum menulis programnya!.

7.3.6 Assesment

Rubrik Penilaian *Problem Solving* 1

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Design input-output-proses	Siswa mampu membuat 3 elemen disain input, proses dan output untuk menghitung keliling persegi	Siswa mampu membuat satu dari 2 elemen disain (input, dan output) untuk menghitung keliling persegi	Siswa mampu membuat satu dari 3 elemen disain (input, dan output) untuk menghitung keliling persegi	Siswa mampu membuat disain input, proses dan output untuk menghitung keliling persegi
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi

Kebenaran Program Menghitung Keliling	Program dapat menghitung keliling dengan benar	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sese kali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

Rubrik Penilaian Problem Solving 2

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Design input-output-proses	Siswa mampu membuat 3 elemen disain input, proses dan output untuk membuat program KTP	Siswa mampu membuat satu dari 2 elemen disain (input, dan output) untuk membuat program KTP	Siswa mampu membuat satu dari 3 elemen disain (input, dan output) untuk membuat program KTP	Siswa mampu membuat disain input, proses dan output untuk membuat program KTP
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program Menghitung Keliling	Program dapat menghitung keliling dengan benar	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sese kali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

7.3.7 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami proses pembuatan program input- output?	
Apakah siswa bisa membuat program input output?	
Apakah siswa mampu mendeteksi kesalahan program yang tidak bisa berjalan?	
Apakah senang belajar bahasa pemrograman pascal?	
Apakah terdapat kesulitan dalam belajar input output dalam bahasa pemrograman pascal?	

7.3.8 Contoh Soal Ulangan

Perhatikan source code program pascal sebagai berikut :

```
Program data_siswa;
Uses wincrt;
Var
Nama,NIS,Alamat:string;
Begin
    Clrscr;
    Write('Masukkan Nama : ');
    readln(Nama);
    Write('Masukkan NIS : ');
    readln(Nim);
    Write('Masukkan Alamat : '); readln(Alamat);
    {Menampilkan hasil input data}
    Writeln;
    Writeln ('Hasil input data');
    Writeln ('Nama : ',Nama);
    Writeln ('Nim : ',NIS);
    Writeln ('Alamat : ',Alamat);
end.
```

- Menurutmu, program di atas apakah sesuai dengan praktik baik penulisan program?
- Tuliskan output dari source code diatas?
- Apabila variabel Nama, NIS dan Alamat diganti dengan tipe lain, bagaimana hasil outputnya, jelaskan! Tipe apa yang mau coba ?

7.4 Aktivitas-4 Assignment

Kode Aktivitas : K10-AP-P3-A4-Assign

Pada aktivitas ini, siswa akan belajar tentang “assignment” (pemberian nilai variabel) dalam bahasa pascal. Keseluruhan aktivitas akan memerlukan 1 pertemuan selama 3 JP.

Materi yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pembuatan program dengan berbagai jenis materi diatas menggunakan free pascal atau ideone.com.

7.4.1 Pertanyaan Pemantik

Pernahkah kalian melakukan transaksi transfer uang menggunakan HP? Apa saja yang harus kalian lakukan? Pasti pertama kalian memasukkan akun dan *password*, kemudian memilih pilihan transaksi apa yang akan dilakukan, setelah itu akan terjadi proses sesuai dengan masukan yang diberikan, dan akhirnya akan mengeluarkan bukti transaksi. Pada saat transaksi transfer uang tersebut terjadi input – proses transaksi sesuai pilihan, dan output. Tahukah kalian jika aplikasi tersebut dibuat dengan menggunakan bahasa pemrograman?

7.4.2 Kata Kunci

assignment (:=), input, output

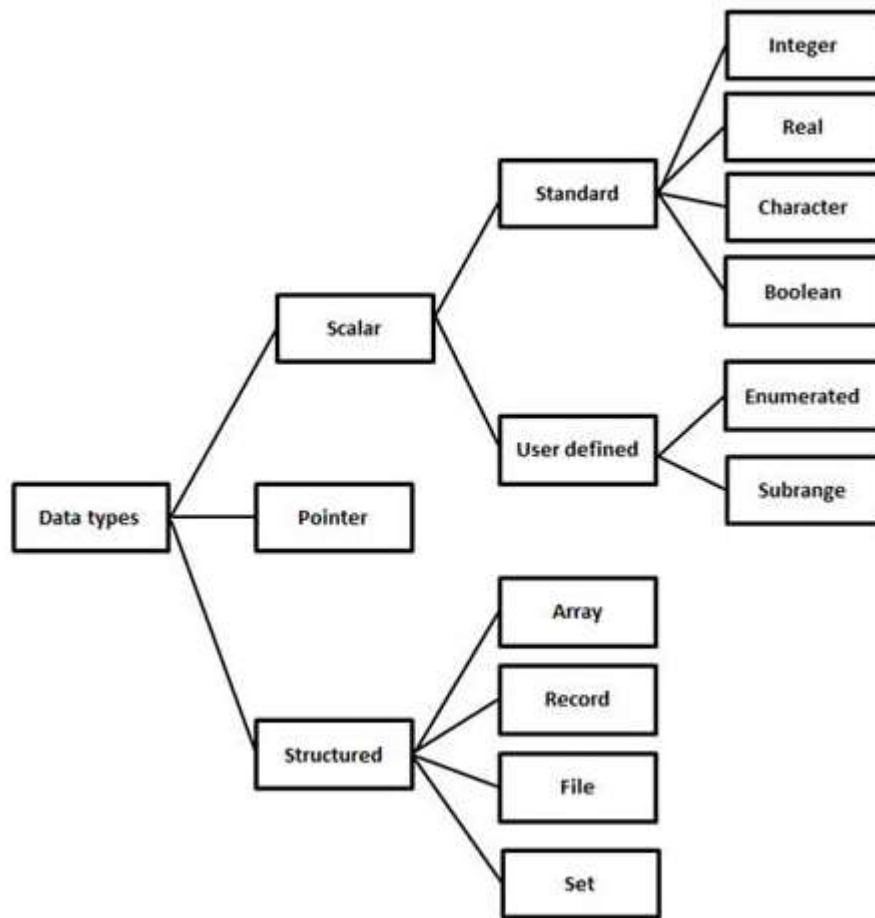
7.4.3 Konsep Terkait Aktivitas

Setelah kita belajar tentang input output pada program pascal, materi yang akan kita pelajari adalah tentang type, variabel, constanta, type, operator aritmatika, operator boolean, operator perbandingan

A. TIPE DATA

Type data adalah atribut dari data yang menyatakan bagaimana data disimpan dan operasi yang dapat dilakukan [https://en.wikipedia.org/wiki/Data_type]. Setiap bahasa pemrograman menyediakan type data dasar yang dapat kita pakai, dan beberapa bahasa menyediakan cara bagaimana membuat type turunan dari type dasar.

Tipe data di dalam bahasa pascal bisa dikelompokkan menjadi berbagai jenis. Gambar berikut bisa menjadi acuan pembagian tipe data ini:



Catatan : pada modul ini hanya dibahas mengenai tipe data skalar yang standar dan array.

A. TIPE DATA STANDAR

Type standar yang disediakan dalam bahasa Pascal adalah integer, real, boolean, char, string. Setiap type variabel dapat menampung nilai sesuai tipenya, dan nilai tersebut hanya dapat diproses untuk operator type terkait.

NO	TIPE DASAR	PENJELASAN	BATASAN NILAI	CONTOH NILAI
1	integer	Bilangan bulat, boleh negatif, Nol atau positif	-2^{15} s.d $2^{15}-1$, atau -32 768 s.d 32 767 (*)	-20 100 0
2	Real	Bilangan riil (boleh mengandung titik desimal), boleh negatif atau positif	$1.5E-45$.. $3.4E38$ (**)	1.55 5.0E2
3	boolean	Bilangan boolean	false .. true	True False

NO	TIPE DASAR	PENJELASAN	BATASAN NILAI	CONTOH NILAI
4	char	Karakter abjad, angka, karakter khusus, kelihatan maupun tidak kelihatan (***)	Karakter sesuai kode ASCII bernilai 0..127	Lihat https://wiki.freepascal.org/ASCII
5	string	Deretan karakter	Setiap karakter mengikuti batasan char	https://wiki.freepascal.org/String

Karena memakai free pascal maka batasan nilai mengacu ke dokumentasi freepascal:

(*) batasan nilai ditentukan oleh compiler, batasan nilai yang ditulis berdasarkan kompiler yang lama. Lihat <https://wiki.freepascal.org/Integer>

(**) E berarti “sepuluh pangkat”. Jadi $5.0E1$ artinya $5.0 * 10^1 = 50$. Lihat <https://www.freepascal.org/docs-html/ref/refsu5.html>

(***) setiap karakter (huruf) diasosiasikan ke kode ascii. Lihat contoh <https://wiki.freepascal.org/ASCII>

Referensi: <https://wiki.freepascal.org/Type>

B. TIPE DATA TERSTRUKTUR dalam Bahasa PASCAL

Kelompok tipe data ketiga adalah tipe data *structured*, atau tipe data terstruktur. Kelompok ini terdiri dari tipe data *Array*, *String*, *Record*, *Set* dan *File*. Pada modul ini, hanya *array* dan *string* yang akan dibahas dan dipakai untuk latihan. *Array* juga akan dibatasi dengan array 1 dimensi (tabel).

B.1. Tipe Data ARRAY

Tipe data *array* adalah tipe data yang terdiri dari kumpulan tipe data lain yang masih setipe. Anggota dari array ini dikenal dengan element. Dalam bahasa pascal, jumlah element *array* harus sudah di tentukan saat array dideklarasikan.

Berikut contoh cara pendefinisian variabel dengan tipe data array berdimensi satu (Tabel) dalam bahasa pascal:

```
var
    nilai: array[0..9] of integer;
```

B.2. Tipe Data STRING

Tipe data *string* adalah tipe data yang bisa menampung banyak karakter sekaligus, seperti kata, atau kalimat. Secara internal di dalam pascal, *string* merupakan array dari tipe data *char*, oleh karena itu, tipe data ini sering dimasukkan ke dalam array.

Cara mendeklarasikan variabel dengan tipe data *string* adalah sebagai berikut :

```
Var
    Nama :string;
```


C. VARIABEL

C.1. Pengertian Variabel

Variabel adalah 'penanda' identitas yang digunakan untuk menampung suatu nilai, dan nilainya dapat diubah. Setiap variabel memiliki nama sebagai identitas variabel tersebut (*identifier*).

Dalam matematika, konsep variabel biasanya menggunakan x atau y , seperti pada persamaan $x = y + 5$

Disini, nilai ' x ' dan ' y ' bisa diisi dengan angka apapun (walaupun dalam persamaan diatas, nilai x bergantung kepada nilai y). Tapi, pengertian "=" dalam matematika berbeda dengan dalam pemrograman

Di dalam pemrograman, nilai variabel bisa berubah, tergantung kebutuhan. Sebagai contoh, apabila kita membuat program menghitung luas lingkaran, kita bisa membuat variabel 'jari2' dan mengisinya dengan nilai '7', kemudian di dalam kode program, kita bisa mengubah nilai variabel jari2 menjadi '8', '10' atau '1000'. Perubahan nilai variabel dapat terjadi karena assignment, atau karena nilainya dibaca melalui perintah `read` atau `readln`.

Referensi : https://wiki.freepascal.org/Variables_and_Data_Types

C.2. Cara Penulisan Variabel di dalam Pascal

Untuk membuat variabel di dalam program pascal, kita harus mendeklarasikannya dalam kepala program. Setiap variabel dalam bahasa Pascal harus ditentukan tipe data yang akan disimpan dalam variabel tersebut, dan sepanjang kode program, variabel tersebut hanya dapat diubah nilainya asalkan masih dalam tipe yang sama. Keharusan mendeklarasikan dan menentukan type variabel sebelum dipakai inilah yang menjadi ciri bahasa ketat type.

Sebagai contoh, jika variabel 'jari2' nilainya berupa 'angka' (numerik), maka dalam bahasa Pascal dapat dideklarasikan bertipe `integer` (bilangan bulat) atau `real` (bilangan riil).

Kita hanya bisa mengisi variabel bertipe `integer` dengan nilai bilangan bulat (angka) seperti 4, 6, atau 90. Kita tidak bisa mengisinya dengan bilangan riil yang mengandung titik desimal, atau dengan huruf atau kata seperti 'A', 'empat', 'lima' atau 'Andi'.

C.3. Aturan Penulisan Variabel dalam Pascal

Sebuah nama variabel dalam bahasa pascal harus ditulis sesuai dengan aturan penamaan identifier (lihat Halaman 22).

- Panjang nama variabel tergantung kepada *compiler* yang digunakan. Beberapa mendukung 32 karakter hingga 63 karakter. Walaupun begitu, sebaiknya gunakan nama variabel yang tidak terlalu panjang.
- Sebelum digunakan, setiap variabel harus dideklarasikan terlebih dahulu.
- Variabel hanya bisa diubah nilainya dengan nilai yang tipenya sama dengan saat variabel tersebut dideklarasikan. Lihat contoh pemberian/perubahan nilai Variabel dengan inisialisasi dan assignment.

C.4. Cara Pendeklarasian Variabel

Sebelum dapat digunakan di dalam kode program, sebuah variabel harus di deklarasikan terlebih dahulu. Berikut format penulisannya:

```
var
    nama_variabel:tipe_data;
```

Perhatikan bahwa penulisan nama variabel dengan tipe datanya dipisah oleh karakter titik dua ":". Diakhir deklarasi juga ditutup dengan karakter titik koma ";;"

Berikut contohnya di dalam kode pascal:

```
var
    nama: string;
    umur: integer;
    alamat: string;
```

Pascal juga membolehkan deklarasi variabel untuk tipe data yang sama dalam 1 baris (dipisahkan dengan tanda koma ", "):

```
var
    nama, alamat: string;
    umur: integer;
```

C.5. Cara Memberikan Nilai Variabel (Assignment)

Setelah variabel dideklarasikan, berikutnya adalah menginput nilai kedalam variabel tersebut. Berikut format penulisannya:

```
nama_variabel := nilai;
```

Perhatikan bahwa untuk memberikan nilai, pascal menggunakan karakter 'titik dua sama dengan' yakni ":=". Di dalam pemrograman, proses pemberian nilai ini dikenal dengan istilah assignment, dan tanda ":=" disebut juga dengan operator assignment di dalam pascal.

Proses *assignment* dibaca dari kanan ke kiri. Perhatikan kode program P-AP-08 berikut ini yang berisi contoh deklarasi dan pengisian nilai variabel yang tersedia dalam bahasa Pascal

```
Program ContohDeklarasi;
(*File : deklarasi.pas*)
(* Contoh deklarasi *)
var
    N: integer;
    Y: real;
    CC: char;
    Found: boolean;
    Nama : string;
begin
    N := 5;
    X := 0.5;
    CC := 'A';
    Found := true;
```

```

        nama:= 'Kaila';
    end.

```

Pada Kode Program P-AP-08, kita memberikan nilai 'Kaila' kedalam variabel nama. Pengisian nilai hanya dapat dilakukan jika typenya cocok. Beberapa contoh yang salah :

```

Program ContohDeklarasi;
(*File : deklarasi1.pas*)
(* Contoh deklarasi *)
var
    N:integer;
    Y:real;
    CC: char;
    Found: boolean;
    Nama : string;
begin
    N := 5.0; //salah, integer diisi real
    Y := 0; // salah, real diisi integer
    CC := N; // salah, char diisi dengan integer
    Found := 1; //salah, boolean diisi integer
    nama:= CC; // salah, string diisi char
end.

```

B.6. Inisialisasi

Pascal juga membolehkan kita untuk menentukan nilai variabel pada saat deklarasi.

```

Program ContohInisialisasi;
(* inisialisasi nilai saat deklarasi *)
var
    N: integer=3;
    Y: real= 0.5;
    CC: char='3';
    Found= true;
    Nama : string= 'Namaku Lily';
begin
    // program yang memakai nama-nama variabel di atas
end.

```

Nilai yang diberikan saat inisialisasi harus sesuai dengan typenya. Berikut ini contoh inisialisasi yang salah.

```

VAR
    Y: real= 8; //salah, variabel bertope real diisi integer
    X:integer= 5.5;//salah, variabel bertype integer diisi. riil
    A: integer = '1'; // salah, kenapa ?

```

Cara memberikan nilai pada saat deklarasi ini dikenal dengan istilah inisialisasi / initialization.

Perhatikan bahwa nilai yang diberikan sesuai dengan type, dan hanya akan dapat dioperasikan sesuai type. Pada contoh di atas, `cc` adalah sebuah karakter (huruf), yang nilainya `'3'` yang berbeda dengan nilai angka 3 yang bertipe `integer`

B.7. Menampilkan Nilai Variabel

Untuk menampilkan nilai variabel, kita tinggal menuliskannya dengan perintah `write` atau `writeln`. Dengan menuliskan namanya, yang akan dituliskan adalah nilainya., Pada potongan program P-AP-09 berikut ini:

```
Program Tulis;
(*File : tulisnilai.pas*)
(* contoh write dan writeln *)
var
    Nama : string= 'Kaila';
    Umur : integer = 30;
begin
    write(nama); //akan ditampilkan 'Kaila'
    writeln(umur); //akan ditampilkan 30
end.
```

D. KONSTANTA (CONSTANT)

D.1. Pengertian Konstanta

Secara singkat, konstanta adalah sebuah nama yang nilainya bersifat tetap dan tidak dapat diubah sepanjang kode program. Umumnya konstanta digunakan untuk nilai yang tidak akan berubah, seperti nilai pi dalam matematika yang bernilai 3.14, atau kecepatan_cahaya yang bernilai 299.792.458 m/detik.

Referensi : <https://wiki.freepascal.org/Constants>

D.2. Cara Penulisan Konstanta di dalam Pascal

Di dalam pascal, sebuah konstanta hanya dapat diisi dengan tipe data dasar, yakni `char`, `integer`, `real`, `boolean`, serta tipe data `string` dan `set`. Penulisan nama konstanta juga mengikuti aturan penulisan identifier (Lihat di halaman 22).

Konstanta tidak bisa diubah nilainya sepanjang kode program.

Berbeda dengan variabel, konstanta **harus ditentukan** nilainya pada saat deklarasi. Berikut format dasar penulisan konstanta:

```
const
    nama_konstanta=nilai_konstanta;
```

Berikut contoh penulisannya di dalam kode program pascal:

```
const
    phi = 3.14; //phi bertipe real
    dollar = 14000; //dollar bertipe integer
    nama = 'Sinta'; //nama bertipe string
```

Perhatikan bahwa pascal menggunakan tanda sama dengan ' = ' untuk menentukan nilai konstanta saat dideklarasikan (proses inisialisasi). **Type dari konstanta ditentukan oleh nilai yang diisikan pada deklarasinya.**

E. OPERATOR DALAM PASCAL

Sebelum membahas jenis-jenis operator dalam bahasa pascal, terdapat istilah ekspresi, operan dan operator.

Sebuah ekspresi terdiri dari gabungan operan dan operator, yang akan dihitung hasilnya oleh komputer sesuai dengan spesifikasi yang diberikan.

Operan adalah nilai yang digunakan di dalam sebuah operasi. Sedangkan Operator adalah suatu "fungsi" khusus yang akan melakukan perhitungan terhadap operan yang diberikan. Operator mempunyai "presedensi" yaitu urutan perhitungan sesuai prioritas, yang aturannya ditentukan oleh kompiler. Untuk menghindari urutan perhitungan yang tidak terduga karena kita tidak membaca cermat spesifikasi kompiler, penulisan ekspresi dapat memakai tanda kurung, yang menolong kita untuk menentukan presedensi.

Sebagai contoh, pada ekspresi: $10 + 2$. Angka 10 dan 2 disebut sebagai operand, sedangkan tanda tambah ('+') adalah operator. Hasil perhitungannya adalah penjumlahan aritmatika bilangan bulat karena 10 dan 2 adalah bilangan bulat.

Dalam bahasa Pascal, semua operan dalam sebuah ekspresi harus sama type-nya (karena bahasa Pascal adalah bahasa ketat tipe), dan hasilnya adalah tipe tersebut. Sebagai contoh : $10+5.5$ akan salah sebab operan pertama adalah bilangan bulat, sedang operan kedua adalah bilangan riil.

Contoh penggunaan tanda kurung : $((3-5)*(4+2))$ akan berbeda hasilnya dengan $3-5*4+2$ Dengan tanda kurung, prioritas ditentukan oleh penulis program. Sangat disarankan dalam menuliskan ekspresi, kita menuliskan dengan tanda kurung lengkap.

Referensi: <https://wiki.freepascal.org/Operator>

Berikut jenis-jenis operator dalam bahasa pemrograman pascal:

E.1. Operator Assignment

Assignment adalah operator yang digunakan untuk memberikan nilai ke dalam suatu variabel.

Di dalam pascal hanya terdapat 1 operator assignment, yakni ':='

Di ruas kiri operator "[:=" harus hanya ada satu nama variabel yang sudah dideklarasikan. Ruas kanan dapat berupa variabel, suatu nilai sesuai type variabel atau ekspresi yang hasilnya juga sesuai dengan type variabel tersebut

Contoh

```
var
    N, M: integer;
    X, Y: real ;
    C: char;
    mystr : string ;
begin
    N := 5;
```

```

M := N * 10;
X := 10.0 * 5.5 - 0.8;
Y := X/99.;
C := 'X';
mystr:="Aku Belajar Pascal"
end.

```

E.2. Operator Aritmatika

adalah operator yang biasa kita temukan untuk operasi matematika. Berikut jenis-jenis operator aritmatika di dalam pascal :

NO	OPERATOR	PENJELASAN	CONTOHEKSPRES I	HASIL PERHITUNGAN
1	+	Pejumlahan	2 + 3	5
2	-	Pengurangan	5 - 2	3
3	*	Perkalian	2 * 3	6
4	/	Pembagian (real/pecahan)	16 / 4	4
5	div	Pembagian (integer/angka bulat)	14 div 4	3
6	mod	Modulo, sisa pembagian bulat	10 div 3	1

E.3. Operator String

Di dalam pascal, hanya terdapat 1 jenis operator string, yakni tanda tambah ' + ' yang digunakan untuk menyambung string (*concatenate*).

Contoh : "belajar pascal" + "di SMA N 2 Playen"

E.4. Operator Perbandingan / Relasional

digunakan untuk membandingkan 2 buah angka, apakah angka tersebut sama besar, lebih kecil, lebih besar, dll. Nilai dari operator perbandingan ini adalah True atau False (boolean).

Berikut operator perbandingan dalam pascal :

NO	OPERATOR	PENJELASAN	CONTOH	HASIL
1	=	Sama dengan	5 = 5	TRUE
2	<>	Tidak sama dengan	5 <> 5	FALSE
3	>	Lebih besar	5 > 6	FALSE
4	<	Lebih kecil	5 < 6	TRUE
5	>=	Lebih besar atau sama dengan	5 >= 3	TRUE

E.5. Operator Logika / Boolean

digunakan untuk menghasilkan nilai boolean true atau false dari 2 kondisi.

Berikut operator logika dalam pascal :

NO	OPERATOR	PENJELASAN	CONTOH EKSPRESI	HASIL
1	and	Akan menghasilkan TRUE jika kedua operand TRUE	TRUE and FALSE	FALSE
2	Or	Akan menghasilkan TRUE jika salah satu operand TRUE	TRUE or FALSE,	TRUE
3	xor	Akan menghasilkan TRUE jika kedua operand berbeda	TRUE xor FALSE,	TRUE
4	not	Negasi: menghasilkan TRUE jika operan bernilai FALSE, atau akan menghasilkan FALSE jika operan bernilai TRUE	not TRUE	FALSE

7.4.4 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN						
Pendahuluan	Pada awal pembelajaran, guru memberikan materi tentang apa itu tipe, variabel, konstanta dan operator dalam pascal. Kemudian guru meminta siswa untuk membuka ideone.com untuk menunjukkan bagaimana mempraktikkan kode program input output.						
Inti	<p>Untuk selanjutnya dalam kegiatan ini siswa akan diajak belajar bagaimana melakukan operasi <i>assignment</i> yaitu sebuah operator yang digunakan untuk memberikan nilai ke dalam suatu variabel. Di dalam pascal hanya terdapat 1 operator assignment, yakni ‘:=’</p> <p>LATIHAN ASSG-01 :Program Assignment (1)</p> <pre> program asign; (* File : ASIGN.PAS *) (* Assigntment dan print *) var(* Kamus *) i : integer; begin (*Algoritma *) i := 5; writeln ('Ini nilai i : ',i); readln; end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang pada LATIHAN ASSG-01 dengan kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td>Var i : integer</td><td>Deklarasi variabel i bertipe data integer</td></tr> <tr> <td> i := 5;</td><td>Perintah mengisi variabel i dengan nilai 5</td></tr> </tbody> </table>	KODE PROGRAM	MAKNA	Var i : integer	Deklarasi variabel i bertipe data integer	i := 5;	Perintah mengisi variabel i dengan nilai 5
KODE PROGRAM	MAKNA						
Var i : integer	Deklarasi variabel i bertipe data integer						
i := 5;	Perintah mengisi variabel i dengan nilai 5						

KEGIATAN	DISKRIPSI KEGIATAN											
	<pre>writeln ('Ini nilai i : ',i);</pre>	Perintah untuk menampilkan teks 'Ini nilai i : 5' yaitu nilai yang disimpan dalam i										
	Setelah selesai latihan dan memahami tentang kode program yang pertama tentang assignment , maka akan melanjutkan latihan program berikutnya.:											
	<p>LATIHAN ASSG-02 : Program Assignment (2)</p> <pre>program asign1; uses crt; (* File : ASIGN1.pas *) (* Assignment, mengetahui nilai min dan max integer *) (* Kamus *) var i : integer; ii : longint; begin (* Algoritma *) i := 1234; ii := 123456 ; writeln ('Ini nilai i=1234 = : ',i); writeln ('Ini nilai ii=123456 : ',ii); writeln ('Ini nilai max integer: ',maxint); writeln ('Ini nilai max longint: ',maxlongint); readln; end.</pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah pada program LATIHAN ASSG-02, dengan kesimpulan sebagai berikut :</p>											
	<table><tr><th>KODE PROGRAM</th><th>MAKNA</th></tr><tr><td>Var i : integer; ii : longint;</td><td>Terdapat deklarasi variabel i dan ii dengan tipe data longinteger</td></tr><tr><td>Writeln ('hello');</td><td>Menampilkan hello ke layar</td></tr><tr><td>i := 1234; ii := 123456 ;</td><td>Perintah mengisi: nilai i dengan 1234 nilai ii dengan 123456</td></tr><tr><td>writeln('Ini nilai i=1234: ',i);</td><td>Perintah untuk menampilkan teks 'Ini nilai i = 1234'</td></tr></table>	KODE PROGRAM	MAKNA	Var i : integer; ii : longint;	Terdapat deklarasi variabel i dan ii dengan tipe data longinteger	Writeln ('hello');	Menampilkan hello ke layar	i := 1234; ii := 123456 ;	Perintah mengisi: nilai i dengan 1234 nilai ii dengan 123456	writeln('Ini nilai i=1234: ',i);	Perintah untuk menampilkan teks 'Ini nilai i = 1234'	
KODE PROGRAM	MAKNA											
Var i : integer; ii : longint;	Terdapat deklarasi variabel i dan ii dengan tipe data longinteger											
Writeln ('hello');	Menampilkan hello ke layar											
i := 1234; ii := 123456 ;	Perintah mengisi: nilai i dengan 1234 nilai ii dengan 123456											
writeln('Ini nilai i=1234: ',i);	Perintah untuk menampilkan teks 'Ini nilai i = 1234'											

KEGIATAN	DISKRIPSI KEGIATAN	
	writeln ('Ini nilai ii=123456 : ',ii);	Perintah untuk menampilkan teks 'Ini nilai ii = 123456 : 123456' Cermatilah apakah benar terisi dengan 123456
	writeln ('Ini nilai max integer: ',maxint);	Perintah untuk menampilkan teks 'Ini nilai max integer:' diikuti nilai maxint yaitu 32767
	writeln ('Ini nilai max longint: ',maxlongint);	Perintah untuk menampilkan teks 'Ini nilai maxlongint' diikuti nilai variabel maxlongint yaitu 2147483647
<p>LATIHAN ASSG-03 : Koding Operator</p> <p>Untuk selanjutnya agar siswa lebih memahami tentang penerapan operator pada bahasa pemrograman pascal, maka buatlah code program berikut :</p> <pre> Program oprator; (* File : oprator.pas *) (* Contoh pengoperasian variabel bertipe dasar *) (* Kamus *) var Bool1, Bool2, TF : boolean; i, j, hsl : integer; x,y,res : real; (*algoritma *) begin writeln ('Utk program ini, baca teksnya dan tambahkan output'); Bool1 := True; writeln('Bool1 berisi=',Bool1); Bool2 := False; writeln('Bool2 berisi=',Bool2); readln; (** contoh-contoh ekspresi: bukan untuk assignment berulang ulang **) TF := Bool1 And Bool2 ; writeln('Bool1 and Bool2 hasilnya =',TF); TF := Bool1 or Bool2 ; writeln('Bool1 OR Bool2 hasilnya=',TF); TF := Not Bool1 ; writeln('NOT Bool1 hasilnya=',TF); TF := Bool1 Xor Bool2 ; Writeln('Bool1 XOR Bool2 hasilnya=',TF); readln; </pre>		

KEGIATAN	DISKRIPSI KEGIATAN
	<pre> (*operasi numerik *) i := 5; j := 2 ; hsl := i+j; writeln('Hasil Penjumlahan i dan j adalah =',hsl); hsl := i - j; writeln('Hasil Pengurangan adalah =',hsl); hsl := i * j; writeln('Hasil Perkalian i dan j adalah =',hsl); hsl := i div j ; (* pembagian bulat *) writeln('Hasil DIV i dan j adalah =',hsl); hsl := i Mod j ; (* sisa *) writeln('Hasil MOD i dan j adalah =',hsl); readln; (*operasi numerik *) x := 5.0 ; y := 2.0 ; res := x+y; writeln('Hasil X+Y adalah =',res); res := x-y; writeln('Hasil X-Y adalah =',res); res := x/y; writeln('Hasil X/Y adalah =',res); res := x*y; writeln('Hasil X*Y adalah =',res); readln; (*operasional relasional numerik *) TF := i < j; writeln('Hasil i<j adalah =',TF); TF := i > j; writeln('Hasil i>j adalah =',TF); TF := i <= j; writeln('Hasil i<=j adalah =',TF); TF := i >= j; writeln('Hasil i>=j adalah =',TF); TF := i <> y; writeln('Hasil i<>j adalah =',TF); readln; end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada disana, dengan kesimpulan sebagai berikut :</p>

KEGIATAN	DISKRIPSI KEGIATAN	
	KODE PROGRAM	MAKNA
	<pre> Var Bool1, Bool2, TF : Boolean; i, j, hsl : Integer; x,y,res : real; </pre>	<p>Deklarasi variabel :</p> <ol style="list-style-type: none"> 1. Bool1, Bool2, TF dideklarasikan dengan tipe Boolean 2. i, j, hsl dideklarasikan dengan tipe data integer 3. x, y, res dideklarasikan dengan tipe data real
	<pre> writeln ('Utk program ini, baca teksnya dan tambahkan output'); </pre>	Program menampilkan 'Utk program ini, baca teksnya dan tambahkan output'
	<pre> Bool1 := True; writeln('Bool1 berisi=',Bool1); Bool2 := False; writeln('Bool2 berisi=',Bool2); readln; </pre>	Mengisi nilai variabel Bool1 dengan TRUE kemudian menampilkan nilainya. Mengisi nilai variabel Bool2 dengan FALSE dan menampilkan nilainya
	<pre> TF := Bool1 And Bool2 ; writeln('Bool1 and Bool2 hasilnya =',TF); </pre>	Variabel TF diisi dengan hasil perhitungan Bool1 AND Bool2 kemudian hasilnya ditampilkan hasilnya FALSE yaitu hasil ekspresi (TRUE AND FALSE)
	<pre> TF := Bool1 or Bool2 ; writeln('Bool1 OR Bool2 hasilnya=',TF); </pre>	Variabel TF diisi dengan hasil perhitungan Bool1 OR Bool2 kemudian hasilnya ditampilkan hasilnya TRUE yaitu hasil ekspresi (TRUE OR FALSE)
	<pre> TF := Not Bool1 ; writeln('NOT Bool1 hasilnya=',TF); </pre>	Variabel TF diisi dengan hasil perhitungan Bool1 NOT Bool2 kemudian hasilnya ditampilkan hasilnya FALSE yaitu hasil ekspresi (TRUE NOT FALSE)
	<pre> TF := Bool1 Xor Bool2 ; Writeln('Bool1 XOR Bool2 hasilnya=',TF); </pre>	Variabel TF diisi dengan hasil perhitungan Bool1 XOR Bool2 kemudian hasilnya ditampilkan hasilnya TRUE yaitu hasil ekspresi (TRUE XOR FALSE)
	<pre> i := 5; j := 2 ; </pre>	Variabel i (integer) diisi dengan nilai 5 Variabel j (integer) diisi dengan nilai 2
	<pre> hsl := i+j; </pre>	Karena i bernilai 5 dan j bernilai 2 maka i+j akan menghasilkan nilai 7. Program akan menampilkan:

KEGIATAN	DISKRIPSI KEGIATAN	
	<pre>writeln('Hasil Penjumlahan i dan j adalah =',hsl);</pre>	'Hasil Penjumlahan i dan j adalah = 7'
	<pre>hsl := i - j; writeln('Hasil Pengurangan adalah = ',hsl);</pre>	Karena i bernilai 5 dan j bernilai 2 maka i-j akan menghasilkan nilai 3. Program akan menampilkan: 'Hasil pengurangan i dan j adalah = 3'
	<pre>hsl := i * j; writeln('Hasil Perkalian i dan j adalah =',hsl);</pre>	Karena i bernilai 5 dan j bernilai 2 maka i* j akan menghasilkan nilai 10. Program akan menampilkan: 'Hasil Perkalian i dan j adalah = 10'
	<pre>hsl := i div j ; (* pembagian bulat *) writeln('Hasil DIV i dan j adalah =',hsl);</pre>	Karena i bernilai 5 dan j bernilai 2 maka i div j akan menghasilkan nilai 2. Program akan menampilkan: 'Hasil DIV i dan j adalah = 2'
	<pre>hsl := i Mod j ; (* sisa *) writeln('Hasil MOD i dan j adalah =',hsl);</pre>	Karena i bernilai 5 dan j bernilai 2 maka i mod j akan menghasilkan nilai 1. Program akan menampilkan: 'Hasil MOD i dan j adalah = 1'
	<pre>x := 5.0 ; y := 2.0 ;</pre>	Nilai yang disimpan dalam variabel x adalah 5.0 Nilai yang disimpan dalam variabel y adalah 2.0
	<pre>res := x+y; writeln('Hasil X+Y adalah =',res);</pre>	Karena x bernilai 5.0 dan y bernilai 2.0 maka x+y akan menghasilkan nilai 7.0000000000000000E+000 Program akan menampilkan: 'Hasil x+y adalah = 7.0000000000000000E+000'
	<pre>res := x-y; writeln('Hasil X-Y adalah =',res);</pre>	Karena x bernilai 5.0 dan y bernilai 2.0 maka x-y akan menghasilkan nilai 3.0000000000000000E+000 Program akan menampilkan: 'Hasil x-y adalah = 3.0000000000000000E+000'
	<pre>res := x/y; writeln('Hasil X/Y adalah =',res);</pre>	Karena x bernilai 5.0 dan y bernilai 2.0 maka x/y akan menghasilkan nilai 2.5000000000000000E+000

KEGIATAN	DISKRIPSI KEGIATAN	
		Program akan menampilkan: 'Hasil x/y adalah = 2.5000000000000000E+000'
	<pre>res := x*y; writeln('Hasil X*Y adalah =',res);</pre>	Karena x bernilai 5.0 dan y bernilai 2.0 maka x-y akan menghasilkan nilai 1.0000000000000000E+000 Program akan menampilkan: 'Hasil x-y adalah = 1.0000000000000000E+000'
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisi pada form yang sudah dibuat	

7.4.5 Lembar Kerja Siswa

(1) LKS problem solving – Pengolahan Nilai Ujian

Ibu Walikelas biasanya mengolah nilai dengan menggunakan excel. Akibat sebuah virus yang sangat ganas, tiba-tiba terjadi masalah dengan aplikasi lembar kerja di seluruh dunia, yang membutuhkan waktu lama untuk memperbaikannya.

Ibu Walikelas memerlukan sebuah program yang akan membantunya mengolah nilai ujian Matematika, Bahasa Indonesia, Bahasa Inggris, Fisika, Kimia, Biologi. Setiap ujian di sekolah diberi nilai antara 0 s.d. 100 (inklusif). Nilai-nilai per siswa sudah dikumpulkan agar semua siswa segera mendapatkan hasilnya. Setiap siswa dicatat nomor dan namanya.

Buatlah kerangka solusi programnya, dan tuliskanlah program pascalnya.

7.4.6 Assesment

Rubrik penilaian ini digunakan untuk memberikan penilaian pada Lembar Kerja Siswa

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Design input-output-proses	Siswa mampu membuat 3 elemen disain input, proses dan output untuk persoalan yang diberikan	Siswa mampu membuat satu dari 2 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat satu dari 3 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat disain input, proses dan output untuk persoalan yang diberikan
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi

Kebenaran Program Menghitung Nilai Siswa	Program dapat menghitung keliling dengan benar	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sese kali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

7.4.7 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami dan dapat menerapkan perintah assignment dan program ?	
Apakah siswa bisa menerapkan operator pascal untuk menghasilkan output program ?	
Apakah siswa mampu mendeteksi kesalahan program yang tidak bisa berjalan?	
Apakah senang belajar bahasa assignment dan operator pemrograman pascal?	
Apakah terdapat kesulitan dalam belajar dalam materi tentang assignment dan operator bahasa pemrograman pascal?	

7.4.8 Contoh Soal Ulangan

Perhatikan Kode Program P-AP-10 berikut :

```

program Luas_Lingkaran;
uses crt;
const
    pi=3.14;
var
    r          : real;
    luas       : real;
begin
    clrscr;
    write ('Masukan Jari-jari = ');read(r);
    luas:=pi*r*r;
    readln;
    writeln('Luas      = ',luas:4:2);
end.

```

Dari program diatas jelaskan maksud dari setiap baris perintah dengan mengisi tabel berikut:

KODE PROGRAM	ARTI KODE PROGRAM

7.5 Aktivitas-5 Struktur Percabangan

Kode Aktivitas : K10-AP-P4-A5-AnalisisKondisi

Pada aktivitas ini, siswa akan belajar tentang percabangan dalam bahasa pascal. Keseluruhan aktivitas diperkirakan akan memerlukan 1 pertemuan selama 3 JP.

Materi yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pembuatan program dengan model percabangan menggunakan free pascal atau ideone.com.

7.5.1 Pertanyaan Pemantik

Pernahkah kalian menggunakan aplikasi gojek untuk memesan sebuah makanan, bagaimana cara pembayaran yang kalian lakukan? Disana terdapat **pilihan** opsi pembayaran , apakah akan menggunakan go pay atau dibayar manual? Jika dibayar dengan menggunakan go pay akan terdapat potongan ongkos kirimnya juga. Taukah kalian bahwa opsi jenis pembayaran itu merupakan salah satu keputusan yang akan dikerjakan oleh sebuah program, berdasarkan pilihan pengguna.

7.5.1 Kata Kunci

Percabangan, kondisi, `if then`, `if then else`, `case`

7.5.2 Konsep Terkait Aktivitas

Instruksi percabangan erat hubungannya dengan dekomposisi pada Berpikir Komputasional. Program yang mengandung percabangan mengatur kode program yang akan dieksekusi berdasarkan evaluasi kondisi.

A. Struktur Percabangan

Perintah kode pemrograman pascal yang digunakan untuk membuat percabangan kode program atau dikenal juga dengan struktur kondisi / struktur logika terdapat kondisi IF THEN, IF THEN ELSE dan struktur CASE.

Referensi: <https://wiki.freepascal.org/IF>

A.1. Konsep Dasar Percabangan Kondisi IF THEN

Konsep dasar dari percabangan perintah **IF THEN** dalam bahasa pemrogram Pascal adalah sebagai berikut:

```
IF (kondisi) THEN
```

```
begin
    (blok kode program)
end;
```

Kondisi berperan sebagai penentu dari struktur percabangan ini. Jika kondisi terpenuhi (menghasilkan nilai *TRUE*), kode program akan dijalankan. Jika kondisi tidak terpenuhi (menghasilkan nilai *FALSE*), tidak terjadi apa-apa.

Kondisi adalah sebuah ekspresi yang hasilnya nilai boolean, yaitu sebuah ekspresi boolean atau ekspresi perbandingan.

Bagian yang ditandai dengan *begin* dan *end*; merupakan “blok” kode program yang akan dijalankan seandainya kondisi bernilai *TRUE*. Setelah itu, Pascal akan lanjut mengeksekusi kode program dibawahnya.

A.2. Konsep Dasar Percabangan Kondisi IF THEN ELSE

Pada dasarnya, kondisi **IF THEN ELSE** adalah tambahan dari kondisi **IF THEN**. Bagian **ELSE** digunakan untuk menjalankan kode program apabila hasil ekspresi kondisi bernilai **FALSE**, atau tidak terpenuhi.

Konsep dasar dari percabangan **IF THEN ELSE** dalam bahasa pemrograman Pascal adalah sebagai berikut:

```
IF (kondisi) THEN
begin
    (blok kode program 1)
end
ELSE (* NOT kondisi yang tidak perlu dikode *)
begin
    (blok kode program 2)
end;
```

Jika kondisi terpenuhi, program pascal akan menjalankan (kode program 1), jika tidak yang akan dijalankan adalah (kode program 2). Perhatikan bahwa tidak ada “;” di antara “end ELSE”

A.3. Percabangan Banyak kasus

Perintah **IF-THEN-ELSE** hanya dapat menangani dua kasus komplementer (yang satu merupakan negasi dari lainnya) saja, yaitu di mana kondisi bernilai *TRUE* dan *FALSE*. Percabangan program ditentukan dengan hanya mengevaluasi “kondisi”. Pada beberapa persoalan, program harus menangani beberapa kasus. Misalnya kita harus menentukan ukuran baju yang ukurannya adalah “X”, “M”, “L”. Bahkan mungkin, ukuran baju bisa lebih dari itu: “XXXL”, “XXL”, “XL”, “L”, “S”, “XS”. Atau, kita perlu menentukan label nama Hari dalam seminggu yang nilainya 1 s.d. 7 menjadi “Senin” s.d. “Minggu”. Kita dapat memanfaatkan **IF-THEN-ELSE** untuk menuliskan program yang percabangannya banyak, dengan menuliskan kode program sebagai berikut, dengan catatan bahwa Blok kode program harus dituliskan di antara “Begin” dan “end”


```

IF (kondisi1) THEN
    begin
        (Blok kode program 1)
    end
ELSE IF (kondisi2) THEN
    begin
        (Blok kode program 2)
    end
ELSE IF (kondisi3) THEN
    begin
        (Blok kode program 3)
    end;

```

Alur eksekusi potongan program tersebut adalah :

- Jika hasil dari kondisi 1 bernilai **TRUE**, maka yang dijalankan adalah (Blok kode program 1),
- Jika tidak dan kondisi 2 bernilai **TRUE**, maka yang akan dijalankan adalah (Blok kode program 2),
- Jika tidak dan kondisi 3 bernilai **TRUE**, maka yang akan dijalankan adalah (Blok kode program 3)

Perhatikanlah bahwa tidak ada “,” di antara **ELSE IF**

A.4. Percabangan CASE

Secara sederhana, struktur percabangan **CASE** mirip seperti struktur **IF THEN ELSE** yang berulang. Jika di dalam **IF THEN ELSE** kita memiliki format penulisan seperti berikut:

```

IF (kondisi1) THEN
    begin
        (blok kode program 1)
    end
ELSE IF (kondisi2) THEN
    begin
        (blok kode program 2)
    end
ELSE IF (kondisi3) THEN
    begin
        (blok kode program 3)
    end;

```

Maka di struktur **CASE**, format penulisannya seperti ini:

```

CASE (variabel) OF
    nilai1 : (blok kode program 1);
    nilai2 : (blok kode program 2);
    nilai3 : (blok kode program 3);
    // dst
end;

```

Walaupun bentuknya mirip, perhatikan beberapa hal mengenai **CASE**:

1. Variabel harus bertipe “ordinal” yaitu `integer`, `char` atau `set` (type set tidak dibahas dalam pemrograman dasar).
2. Eksekusi program akan dilakukan berdasarkan evaluasi sebagai berikut :
 - a. Jika variabel=nilai1, maka eksekusi (blok kode program 1) lalu ke `end`;
 - b. Jika variabel=nilai2, maka eksekusi (blok kode program 2) lalu ke `end`;
 - c. Jika variabel=nilai3, maka eksekusi (blok kode program 3) lalu ke `end`;
3. Nilai1, nilai2, nilai3 dapat berupa list nilai (artinya beberapa nilai), variabel akan dibandingkan apakah sama dengan nilai yang ada pada list. Eksekusinya akan ditunjukkan pada latihan 05.
4. Jika kondisi mengandung operasi yang bukan hanya mengecek kesamaan, maka tidak bisa memakai **CASE**, harus memakai **IF** banyak kasus

Referensi: <https://www.freepascal.org/docs-html/ref/refsu55.html>

7.5.3 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN
Pendahuluan	Guru memberikan pengantar bahwa dalam pembuatan program, ada saatnya kita butuh suatu perintah percabangan, yakni jika sebuah kondisi terpenuhi, jalankan kode program ini. Jika tidak, jalankan kode program yang lain. Guru menjelaskan tentang apa itu percabangan dan jenisnya. Kemudian akan mengerjakan beberapa latihan koding percabangan berdasarkan beberapa kondisi yang berbeda-beda
Inti	<p>Latihan IF-01 : IF tanpa ELSE</p> <p>Untuk yang pertama guru akan mengenalkan dengan IF dengan 1 kondisi. Untuk dapat memahami guru meminta siswa untuk membuka program pascal untuk bisa mencoba mempraktikkan percabangan dengan IF THEN atau IF dengan 1 kondisi, seperti contoh program P-AP-11 berikut ini.</p> <pre> Program IF_Tunggal; uses crt; (* File : IF1.PAS *) (* contoh pemakaian IF satu kasus *) (* membaca nilai integer, menuliskan nilainya jika positif *) var a : integer; begin (* Program *) clrscr; writeln ('Contoh IF satu kasus '); write ('Ketikkan satu nilai integer : '); readln (a); if (a >= 0) then begin writeln ('Nilai a positif... ', a); end; end. </pre>

KEGIATAN	DISKRIPSI KEGIATAN												
	<p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada program P-AP-11, dengan kesimpulan sebagai berikut :</p> <table> <tr> <th>KODE</th><th>MAKNA</th></tr> <tr> <td>Var a : integer</td><td>Deklarasi variabel a bertipe integer</td></tr> <tr> <td>writeln ('Contoh IF satu kasus ');</td><td>Program Menampilkan teks Contoh IF satu kasus</td></tr> <tr> <td>write ('Ketikkan satu nilai integer : ');</td><td>Program menampilkan teks Ketikkan satu nilai integer</td></tr> <tr> <td>readln (a);</td><td>Program meminta input sebuah nilai integer untuk mengisi nilai a yang bertipe integer</td></tr> <tr> <td>if (a >= 0) then begin writeln ('Nilai a positif...', a)</td><td>Jika nilai a > 0 maka ditampilkan Output 'Nilai a positif' diikuti oleh nilai a</td></tr> </table> <p>Latihan IF-02 : IF-THEN-ELSE</p> <p>Setelah selesai latihan dan memahami IF dengan 1 kondisi, maka selanjutnya kita akan belajar dengan program yang akan dieksekusi untuk hasil ekspresi TRUE atau FALSE Siswa diminta untuk mengkode program P-AP-12 berikut :</p> <pre> program IF2; (* File :IF2.PAS *) (* contoh pemakaian IF dua kasus komplementer *) (* Membaca sebuah nilai, *) (* menuliskan 'Nilai a positif , nilai a', jika a >=0 *) (* 'Nilai a negatif , nilai a', jika a <0 *) uses crt; var a : integer; begin (* Program *) writeln ('Contoh IF dua kasus '); write ('Ketikan suatu nilai integer :'); readln (a); if (a >= 0) then begin writeln ('Nilai a positif ', a); end else (* a<0 *) begin writeln ('Nilai a negatif ', a); end; end. </pre>	KODE	MAKNA	Var a : integer	Deklarasi variabel a bertipe integer	writeln ('Contoh IF satu kasus ');	Program Menampilkan teks Contoh IF satu kasus	write ('Ketikkan satu nilai integer : ');	Program menampilkan teks Ketikkan satu nilai integer	readln (a);	Program meminta input sebuah nilai integer untuk mengisi nilai a yang bertipe integer	if (a >= 0) then begin writeln ('Nilai a positif...', a)	Jika nilai a > 0 maka ditampilkan Output 'Nilai a positif' diikuti oleh nilai a
KODE	MAKNA												
Var a : integer	Deklarasi variabel a bertipe integer												
writeln ('Contoh IF satu kasus ');	Program Menampilkan teks Contoh IF satu kasus												
write ('Ketikkan satu nilai integer : ');	Program menampilkan teks Ketikkan satu nilai integer												
readln (a);	Program meminta input sebuah nilai integer untuk mengisi nilai a yang bertipe integer												
if (a >= 0) then begin writeln ('Nilai a positif...', a)	Jika nilai a > 0 maka ditampilkan Output 'Nilai a positif' diikuti oleh nilai a												

KEGIATAN	DISKRIPSI KEGIATAN						
	<p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada program P-AP-12, dengan kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td> <pre>Var i : integer; ii : longint;</pre> </td><td>Deklarasi variabel i bertipe integer, dan ii bertipe data longinteger</td></tr> <tr> <td> <pre>if (a >= 0) then begin writeln ('Nilai a positif ', a); end else (* a<0 *) begin writeln ('Nilai a negatif ', a); end; readln;</pre> </td><td> <p>jika nilai a lebih besar atau sama dengan 0 maka ditampilkan 'Nilai a positif ... diikuti nilai a jika tidak (artinya <0) maka ditampilkan 'Nilai a negatif ... diikuti nilai a</p> </td></tr> </tbody> </table> <p>Latihan 03 – IF dengan Banyak Kasus</p> <p>Setelah selesai latihan dan memahamai tentang IF-THEN dan IF-THEN-ELSE, siswa diajak memahami program yang memerlukan banyak percabangan, di mana setiap potongan program menangani kasus tertentu. Setiap kasus, akan dituliskan sebagai satu kondisi. Kita memanfaatkan “ELSE” untuk mengatur percabangan.</p> <p>Siswa diminta untuk “coding” kode program P-AP-13 berikut :</p> <pre>(* File : IF3.PAS *) (* contoh pemakaian IF dua kasus komplementer *) (* Membaca sebuah nilai, *) (* menuliskan 'Nilai a positif , nilai a', jika a>0 *) (* 'Nilai a sama dengan nol , nilai a', jika a =0 *) (*'Nili a negatif , nilai a', jika a <0 *) program IF3; user crt; (* Kamus *) var a : integer; begin (* Program *) clrscr; writeln ('Contoh IF tiga kasus'); write ('Ketikkan suatu nilai integer :'); readln (a); if (a>0) then begin</pre>	KODE	MAKNA	<pre>Var i : integer; ii : longint;</pre>	Deklarasi variabel i bertipe integer, dan ii bertipe data longinteger	<pre>if (a >= 0) then begin writeln ('Nilai a positif ', a); end else (* a<0 *) begin writeln ('Nilai a negatif ', a); end; readln;</pre>	<p>jika nilai a lebih besar atau sama dengan 0 maka ditampilkan 'Nilai a positif ... diikuti nilai a jika tidak (artinya <0) maka ditampilkan 'Nilai a negatif ... diikuti nilai a</p>
KODE	MAKNA						
<pre>Var i : integer; ii : longint;</pre>	Deklarasi variabel i bertipe integer, dan ii bertipe data longinteger						
<pre>if (a >= 0) then begin writeln ('Nilai a positif ', a); end else (* a<0 *) begin writeln ('Nilai a negatif ', a); end; readln;</pre>	<p>jika nilai a lebih besar atau sama dengan 0 maka ditampilkan 'Nilai a positif ... diikuti nilai a jika tidak (artinya <0) maka ditampilkan 'Nilai a negatif ... diikuti nilai a</p>						

KEGIATAN	DISKRIPSI KEGIATAN				
	<pre> writeln ('Nilai a positif ', a); end; else if (a=0) then begin writeln ('Nilai a sama dengan nol ', a); end; else if (a<0) then begin writeln ('Nilai a negatif ', a); end; end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah IF yang ada dalam program P-AP-13, dengan kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td> <pre> if (a>0) then begin writeln ('Nilai a positif ', a); end; else if (a=0) then begin writeln ('Nilai a sama dengan nol ', a); end; else if (a<0) then begin writeln ('Nilai a negatif ', a); end; end; </pre> </td><td> <p>Ada tiga kemungkinan output tergantung dari nilai a. Berikut ini contoh beberapa nilai a. Isilah Outputnya</p> <ul style="list-style-type: none"> • Jika nilai a=10 maka Outputnya: Positif • Jika nilai a=-10 maka Outputnya: Negatif • Jika nilai a=0 maka Outputnya: sama dengan nol • Jika nilai a=100 maka Outputnya: Positif • Jika nilai a=-99 maka Outputnya: Negatif </td></tr> </tbody> </table> <p>Latihan IF-04 : CASE</p> <p>Selanjutnya untuk latihan yang terakhir kita akan menggunakan struktur CASE. Sebenarnya struktur CASE ini fungsinya sama dengan IF THEN ELSE, namun hanya meringkas pernyataan IF. Perhatikan bahwa “kondisi” pada instruksi case tidak dituliskan sebagai ekspresi yang eksplisit, dan variabel yang dibandingkan harus bertipe integer atau karakter. Siswa diminta untuk menuliskan kode program P-AP-14 berikut :</p>	KODE PROGRAM	MAKNA	<pre> if (a>0) then begin writeln ('Nilai a positif ', a); end; else if (a=0) then begin writeln ('Nilai a sama dengan nol ', a); end; else if (a<0) then begin writeln ('Nilai a negatif ', a); end; end; </pre>	<p>Ada tiga kemungkinan output tergantung dari nilai a. Berikut ini contoh beberapa nilai a. Isilah Outputnya</p> <ul style="list-style-type: none"> • Jika nilai a=10 maka Outputnya: Positif • Jika nilai a=-10 maka Outputnya: Negatif • Jika nilai a=0 maka Outputnya: sama dengan nol • Jika nilai a=100 maka Outputnya: Positif • Jika nilai a=-99 maka Outputnya: Negatif
KODE PROGRAM	MAKNA				
<pre> if (a>0) then begin writeln ('Nilai a positif ', a); end; else if (a=0) then begin writeln ('Nilai a sama dengan nol ', a); end; else if (a<0) then begin writeln ('Nilai a negatif ', a); end; end; </pre>	<p>Ada tiga kemungkinan output tergantung dari nilai a. Berikut ini contoh beberapa nilai a. Isilah Outputnya</p> <ul style="list-style-type: none"> • Jika nilai a=10 maka Outputnya: Positif • Jika nilai a=-10 maka Outputnya: Negatif • Jika nilai a=0 maka Outputnya: sama dengan nol • Jika nilai a=100 maka Outputnya: Positif • Jika nilai a=-99 maka Outputnya: Negatif 				

KEGIATAN	DISKRIPSI KEGIATAN				
	<pre> program KASUS; (* File : KASUS.PAS *) (* Contoh kasus dengan intruksi CASE *) var cc : char; begin (*Program*) writeln ('Ketikkan sebuah huruf, akhiri dengan RETURN '); readln (cc); case cc of 'a' : begin writeln (' Yang anda ketik adalah a '); end; 'u' : begin writeln (' Yang anda ketik adalah u '); end; 'e' : begin writeln (' Yang anda ketik adalah e '); end; 'o' : begin writeln (' Yang anda ketik adalah o '); end; 'i' : begin writeln (' Yang anda ketik adalah i '); end else writeln ('Yang anda ketik adalah huruf mati atau angka'); end; end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada kode program P-AP-14, dengan kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td> <pre> case cc of 'a' : begin writeln ('Yang anda ketik adalah a'); end; 'u' : </pre> </td><td> <p>Program hanya akan menampilkan salah satu output, tergantung kepada nilai yang diketikkan dan disimpan dalam variabel a.</p> <ul style="list-style-type: none"> Jika user mengetik ' z ' maka Outputnya : yang anda </td></tr> </tbody> </table>	KODE PROGRAM	MAKNA	<pre> case cc of 'a' : begin writeln ('Yang anda ketik adalah a'); end; 'u' : </pre>	<p>Program hanya akan menampilkan salah satu output, tergantung kepada nilai yang diketikkan dan disimpan dalam variabel a.</p> <ul style="list-style-type: none"> Jika user mengetik ' z ' maka Outputnya : yang anda
KODE PROGRAM	MAKNA				
<pre> case cc of 'a' : begin writeln ('Yang anda ketik adalah a'); end; 'u' : </pre>	<p>Program hanya akan menampilkan salah satu output, tergantung kepada nilai yang diketikkan dan disimpan dalam variabel a.</p> <ul style="list-style-type: none"> Jika user mengetik ' z ' maka Outputnya : yang anda 				

KEGIATAN	DISKRIPSI KEGIATAN
	<pre> begin writeln ('Yang anda ketik adalah u'); end; 'e' : begin writeln ('Yang anda ketik adalah e'); end; 'o' : begin writeln ('Yang anda ketik adalah o'); end; 'i' : begin writeln ('Yang anda ketik adalah i'); end else writeln ('Yang anda ketik adalah huruf mati atau angka'); end; </pre> <p>ketik adalah huruf mati atau angka</p> <ul style="list-style-type: none"> • Jika user mengetik 'i' maka Outputnya :..... • Jika user mengetik 'a' maka Outputnya : yang anda ketik adalah huruf a • Jika user mengetik 'o' maka Outputnya : yang anda ketik adalah huruf o • Jika user mengetik 'e' maka Outputnya : yang anda ketik adalah huruf e • Jika user mengetik 'u' maka Outputnya : yang anda ketik adalah huruf u • Jika user mengetik 'x' maka Outputnya : yang anda ketik adalah huruf mati atau angka

Latihan IF-05 : CASE dengan list nilai

Misalnya kita akan menulis program jika b bernilai 1 atau 7 atau 2037 atau 5; dan untuk bukan dalam list nilai tersebut. Programnya dapat kita tulis sebagai berikut :

```

if (b = 1) or (b = 7) or (b = 2037) or (b = 5)
then
  // Blok kode program1
else
  //Blok kode program2;

```

Program tersebut akan lebih sederhana penulisannya dengan memanfaatkan instruksi case sebagai berikut:

```

case b of
  1,7,2037,5: // Blok kode program1;
  otherwise  // Blok kode program2
end;

```

Silahkan kerjakan latihan berikut, sebagai penerapannya :

KEGIATAN	DISKRIPSI KEGIATAN
	<p>Restoran “Enak” akan menampilkan menu hari ini. Setiap hari Senin, Selasa, Rabu restoran Enak menjual nasi gudeg, nasi soto dan mie ayam. Hari Kamis dan Jumat, restoran Enak menjual nasi pecel dan nasi rawon. Sedangkan Sabtu dan Minggu restoran Enak menjual nasi Padang. Nama-nama hari dikode menjadi nilai 1 s.d. 7, dengan 1 adalah Senin, dan 7 adalah Minggu.</p> <p>Jawab :</p> <pre> Program menu; uses crt; (*File:restoranenak.pas*) (* program mencetak menu hari ini restoran Enak *) VAR kodeHari: integer; Begin clrscr; write('Silahkan input menu hari ke(1-7): '); readln(kodeHari); case kodeHari of 1,2,3: begin writeln ('nasi gudeg, nasi soto, mie ayam'); readln; end; 4,5 : begin writeln ('nasi pecel, nasi rawon'); readln; end; 6,7 : begin writeln ('nasi Padang'); readln; end; otherwise // Blok kode program2 end; end. </pre>
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisikan pada form yang sudah dibuat

Lembar Kerja Siswa

(1) LKS progcil

Buatlah sebuah program yang membaca sebuah nilai siswa, dan menentukan predikatnya berdasarkan ketentuan berikut :

- Jika nilai rentang 0-50 nilai siswa maka memunculkan predikat E
- Jika nilai rentang 51-70 nilai siswa maka memunculkan predikat D

- c. Jika nilai rentang 71-80 nilai siswa maka memunculkan predikat C
- d. Jika nilai rentang 81-90 nilai siswa maka memunculkan predikat B
- e. Jika nilai rentang 91-100 nilai siswa maka memunculkan predikat A

Buatlah rancangan program percabangan menggunakan sistem **IF THEN ELSE** dan **CASE** , setelah yakin benar, tuliskan dan ujilah program yang kamu buat.

(2) LKS problem solving

Latihan ini adalah latihan mengkonstruksi IF yang kompleks. Siswa diminta “merancang” kerangka instruksi IF yang akan dituliskan di kertas, sebelum menuliskan program. Setelah kerangka IF benar, baru diizinkan untuk memprogram. Untuk program yang kompleks, tidak disarankan untuk langsung melakukan koding

Deskripsi Persoalan :

Kadar hemoglobin dalam darah manusia berbeda – beda sesuai kategori kelamin dan usia. Pada bayi yang baru lahir kadar hemoglobinnya adalah 17 sampai 22 gram per desiliter (g/dl), pada anak – anak 11 sampai 13 g/dl, pria dewasa 14 sampai 18 g/dl dan wanita dewasa 12 sampai 16 g/dl.

Tuliskanlah sebuah program yang membaca usia, kode jenis kelamin {pria, wanita} dan akan menentukan apakah seseorang “anemia” atau terlalu rendah, HB nya normal, atau “Terlalu tinggi”.

7.5.4 Assesment

Rubrik Penilaian Progcil pada Lembar Kerja Siswa - (1) LKS Progcil

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program dengan menggunakan sistem IF THEN ELSE dan CASE	Program dapat menerapkan sistem IF THEN ELSE dan CASE	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengkoreksi secara mandiri	Siswa sesekali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

Rubrik Penilaian Problem Solving – (2) LKS Problem Solving

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Mampu membuat notasi algoritma	Mampu membuat rancangan notasi algoritma berupa flowchart dan pseudocode	Mampu membuat rancangan notasi algoritma berupa flowchart saja	Mampu membuat rancangan notasi algoritma berupa pseudocode saja	Tidak mampu membuat rancangan notasi algoritma berupa flowchart dan pseudocode
Design input-output-proses	Siswa mampu membuat 3 elemen disain input, proses dan output untuk persoalan yang diberikan	Siswa mampu membuat satu dari 2 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat satu dari 3 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat disain input, proses dan output untuk persoalan yang diberikan
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program menggunakan sistem IF kompleks	Program dapat menggunakan sistem IF kompleks	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sese kali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

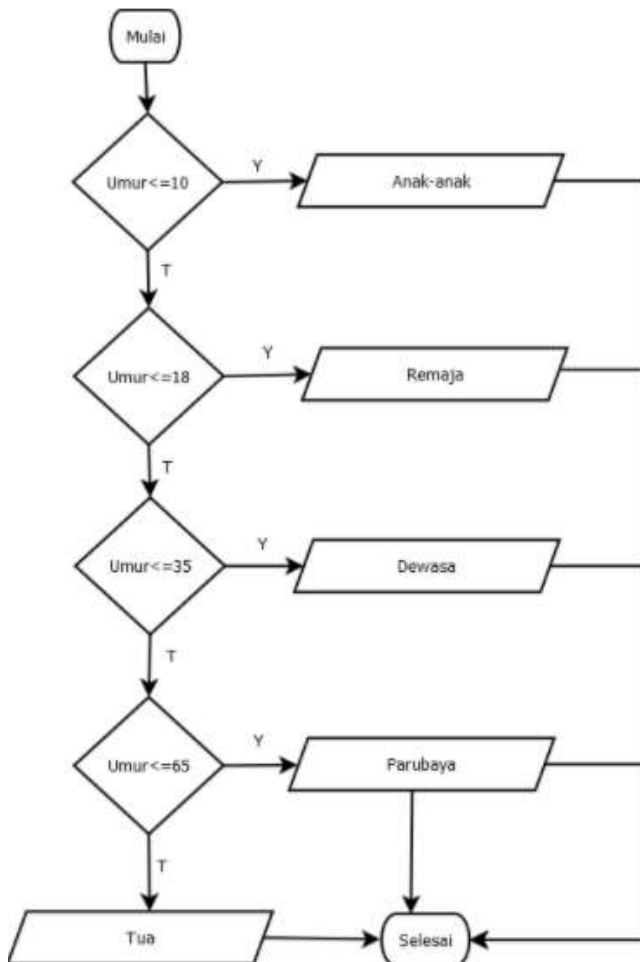
7.5.5 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami dan dapat menerapkan perintah assignment dan program ?	
Apakah siswa bisa menerapkan operator pascal untuk menghasilkan output program yang program?	
Apakah siswa mampu mendeteksi kesalahan program yang tidak bisa berjalan?	
Apakah senang belajar bahasa assignment dan operator pemrograman pascal?	

Aspek	Refleksi Siswa
Apakah terdapat kesulitan dalam belajar dalam materi tentang assignment dan operator bahasa pemrograman pascal?	

7.5.6 Contoh Soal Ulangan

Perhatikan flowchart berikut :



Dari flowchart diatas buatlah kode program untuk mampu mendapatkan output yang sesuai yang diharapkan pada flowchart tersebut.

7.6 Aktivitas-6 Perulangan (loop)

Kode Aktivitas : K10-AP-P5-A6-loop

Pada aktivitas ini, siswa akan belajar tentang perulangan / loop dalam bahasa Pascal. Keseluruhan aktivitas diperkirakan akan memerlukan 1 pertemuan selama 3 JP.

Materi yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pembuatan program dengan berbagai jenis materi diatas menggunakan free pascal atau ideone.com.

7.6.1 Pertanyaan Pemantik

Pernahkah kalian perhatikan ketika kita berada di perempatan lampu merah. Lampu akan bergantian menyala anatar merah, kuning dan hijau. Dan hal tersebut akan diulang-ulang agar kondisi jalan dapat terkontrol sehingga tidak menimbulkan kepadatan bahkan kecelakaan. Taukah kalian bahwa pengaturan waktu perubahan nyala lampu itu merupakan salah satu keputusan perulangan yang akan dikerjakan oleh sebuah program.

7.6.2 Kata Kunci

Loop (perulangan), `For do`, `For Down to`, `while-do`, `repeat... until`

7.6.3 Konsep Terkait Aktivitas

Setelah kita belajar tentang percabangan pada program pascal, materi selanjutnya yang akan kita pelajari adalah tentang perulangan/ loop.

Perulangan atau dalam bahasa inggris dikenal dengan istilah loop merupakan konsep untuk mengulang satu blok kode (satu atau lebih baris perintah). Disini akan dibahas cara membuat perulangan di Pascal menggunakan perintah `FOR TO`, `FOR DOWNTO`, `WHILE DO` dan `REPEAT UNTIL`.

Perulangan atau loop adalah konsep pemrograman dimana kita mengulang baris program beberapa kali. “Beberapa kali” disini berarti 1 atau beberapa kali, namun pemrogram perlu menjamin bahwa program harus berhenti, kecuali program memang dirancang untuk hidup terus menerus misalnya program yang mengendalikan server yang harus hidup. Pada umumnya, program harus berhenti (determinasi).

A. Pengulangan FOR

Pengulangan `FOR` pada hekekatnya adalah pengulangan yang dikendalikan oleh pencacah. Ada dua variasi `FOR`, yaitu dengan pencacah menaik (`FOR TO`), dan dengan pencacah menurun (`FOR DOWNTO`).

Misalnya, kita ingin anda menulis teks “Hello World” sebanyak 1000 kali. Tentu sangat melelahkan mengetik semua ini (walaupun dengan di-copy paste). Menggunakan loopi, kita bisa membuatnya dalam waktu singkat dan cepat, hanya butuh beberapa baris kode program, dan komputer akan mencetak untuk kita

Salah satu struktur perulangan di dalam Pascal adalah `FOR TO`, berikut format penulisannya:

```
FOR (variabel_counter) := (nilai_awal) TO (nilai_akhir) DO
begin
    (blok kode program yang ingin diulang disini...)
end;
```

variabel_counter adalah variabel integer yang berfungsi sebagai counter , atau pencacah di dalam perulangan. Variabel ini otomatis menaik dari nilai_awal hingga nilai_akhir dengan pertambahan 1. Dalam setiap perulangan, blok kode program yang berada di dalam begin dan end; akan dijalankan. variabel_counter ini bisa digunakan namun tidak disarankan untuk

dipakai buat keperluan lain karena berpotensi mengganggu jalannya pengulangan jika berubah dalam badan loop.

Syarat agar pengulangan dapat dilakukan, `nilai_awal <= nilai_akhir`

Perulangan **FOR DOWNTO**

Perulangan **FOR DOWNTO** pada dasarnya sangat mirip dengan perulangan **FOR TO**, bedanya perulangan ini khusus untuk iterasi yang menurun, dengan nilai pencacah yang mengecil mulai `nilai_awal` s.d. `nilai_akhir`.

Berikut format penulisannya:

```
FOR (variabel_counter) := (nilai_awal) DOWNTO (nilai_akhir) DO
begin
    (* blok kode program yang ingin diulang disini...*)
end;
```

Syarat agar pengulangan dapat dilakukan, `nilai_awal >= nilai_akhir`

Referensi : <https://wiki.lazarus.freepascal.org/For>

B. Perulangan **WHILE DO**

Perulangan **FOR TO** dan **FOR DOWNTO** yang sudah dibahas sebelumnya cocok untuk kondisi dimana kita sudah tahu berapa banyak perulangan yang ingin dijalankan. Dalam **FOR**, nilai awal perulangan dan nilai akhir sudah harus ditentukan sebelum instruksi **FOR**.

Untuk situasi dimana banyaknya perulangan belum bisa dipastikan, kita bisa menggunakan perulangan **WHILE DO** atau **REPEAT UNTIL**, yang perulangannya akan berhenti bukan karena pencacah tetapi karena suatu kondisi dipenuhi

Berikut format dasar penulisan perulangan **WHILE DO** dalam bahasa pemrograman PASCAL:

```
WHILE (kondisi-ulangi) DO
begin
    (*blok kode program yang ingin diulang...*)
    (*blok kode program untuk mengubah condition...*)
end;
```

Kunci dari perulangan **WHILE DO** ada di kondisi dan blok kode program untuk mengubah kondisi. Kondisi adalah ekspresi boolean atau ekspresi perbandingan, yang harus bernilai **TRUE** agar perulangan bisa dijalankan. Selama kondisi terpenuhi (bernilai **TRUE**), perulangan akan terus dijalankan. Jika kondisi tidak terpenuhi (bernilai **FALSE**), perulangan tidak akan berjalan.

Referensi : <https://wiki.freepascal.org/While>

C. Perulangan **REPEAT UNTIL**

Pada dasarnya, perulangan `REPEAT UNTIL` mirip seperti perulangan `WHILE DO`, dimana kita akan melakukan melakukan pengulangan selama suatu kondisi masih dipenuhi.

Perbedaan antara pengulangan `REPEAT UNTIL` dengan `WHILE DO`:

1. Pada `REPEAT UNTIL`, pemeriksaan kondisi ini dilakukan di **akhir perulangan** dan kondisi yang dicek adalah **kondisi berhenti**, bukan di awal seperti `WHILE DO` yang mengecek kondisi pengulangan. Catatan : kondisi berhenti adalah negasi dari kondisi pengulangan.
2. Blok kode pada blok kode yang diulang pada `REPEAT UNTIL` minimal pasti pernah dieksekusi satu kali karena pengecekan kondisi dilakukan setelah blok kode dijalankan. Sedangkan badan pengulangan pada `WHILE DO` ada kemungkinan tidak pernah dieksekusi, yaitu pada jika saat sebelum memasuki loop pertama kali kondisi bernilai `FALSE`.



Berikut format dasar penulisan perulangan `REPEAT UNTIL` dalam bahasa pemrograman PASCAL:

```
REPEAT
  begin
    (*blok kode program yang ingin diulang disini...*)
    (*blok kode program yang untuk mengubah condition...*)
  end;
UNTIL (*kondisi-berhenti*)
```

Referensi: <https://wiki.freepascal.org/REPEAT..UNTIL>

7.6.4 Gambaran Umum Kegiatan

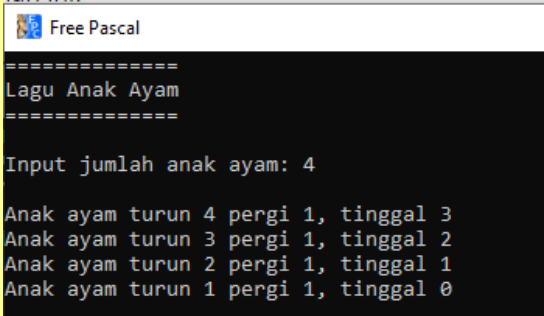
KEGIATAN	DISKRIPSI KEGIATAN
Pendahuluan	Pada awal pembelajaran, guru menjelaskan perulangan atau looping dalam pemrograman pascal. Terdapat 3 jenis perulangan dalam pascal yaitu <code>FOR</code> , <code>WHILE DO</code> dan <code>REPEAT UNTIL</code> . Untuk lebih memahami dari setiap jenis perulangan guru mengajak siswa untuk berlatih membuat program perulangan.
Inti	Latihan loop-01 : FOR Untuk yang pertama, siswa diminta untuk coding dari contoh penerapan perulangan <code>FOR DO</code> sebagai berikut. Bacalah dengan cermat contoh Program P-AP-15 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) : <pre>Program for_do; (*File :fordo.pas*) (*looping dengan for_do*) var i: integer; begin for i := 1 to 1000 do</pre>

KEGIATAN	DISKRIPSI KEGIATAN
	<pre>begin writeln('Hello World'); end; readln; end.</pre> <div data-bbox="434 459 702 1155">  </div> <div data-bbox="730 459 1380 1265"> <p>Tulisan Hello World akan diulang sebanyak 1000 kali sesuai perulangan yang dibuat pada program.</p> <p>Perintah yang membuat tulisan hello word bisa muncul 1000 kali adalah pada kode program berikut :</p> <pre>for i := 1 to 1000 do begin writeln('Hello World'); end;</pre> <p>Untuk membuat perulangan sebanyak 1000 kali, kita menggunakan perintah for i := 1 to 1000 do.</p> <p>Ini bisa dibaca dengan “Untuk variabel i, jalankan perulangan mulai dari 1,2,3,4,s.d 1000”.</p> <p>Apa yang akan diulang? Adalah blok program yang diawali perintah begin, dan diakhiri perintah end;. Di dalam blok ini kita membuat sebuah perintah: writeln('Hello World').</p> </div> <p>Setelah selesai latihan guru menanyakan kepada siswa bagaimana jika siswa akan memunculkan outputnya sebagai berikut :</p> <div data-bbox="434 1384 646 1957">  </div>

KEGIATAN	DISKRIPSI KEGIATAN
	<p>Didalam output tersebut akan tertulis urutan hellow world dari Hello World ke-1 sampai dengan Hello world ke-1000.</p> <p>Setelah selesai guru meminta siswa memperlihatkan hasilnya di depan kelas sebagai perwakilan.</p> <p>Selanjutnya setelah selesai membahas tentang perulangan FOR DO selanjutnya akan dibahas tentang perulangan DOWN TO. Guru meminta kembali siswa untuk membuat program berikut :</p> <p>Latihan loop-02 : DOWNT0</p> <p>Bacalah dengan cermat contoh Program P-AP-16 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre> Program down_to; (*File :downto.pas*) (*Menggunakan fungsi loop down to untuk menghitung mundur dari 10,9 8,...1*) uses crt; var i: integer; begin clrscr; for i := 10 downto 1 do begin writeln('Hitung mundur: ',i); end; readln; end. </pre> <p>Setelah menyelesaikan 2 jenis latihan FOR DO dan DOWNT0 guru bertanya kepada siswa, apa perbedaan mendasar dari 2 jenis perulangan tersebut?</p> <p>Siswa berdiskusi kemudian menyimpulkan bahwa perbedaan mendasar dari keduanya adalah apabila perulangan :</p> <ol style="list-style-type: none"> 1. FOR TO digunakan untuk perulangan dari angka terkecil ke terbesar, namun untuk DOWN TO digunakan untuk perulangan dari angka terbesar ke angka terkecil 2. Perulangan menggunakan FOR TO dan DOWN TO harus diketahui terlebih dahulu pembatasan perulangannya. <p>Latihan loop-03 : WHILE DO</p> <p>Untuk selanjutnya akan dipelajari jenis perulangan dengan menggunakan WHILE DO, guru mengajak siswa untuk menuliskan kode program P-AP-17 yang akan menuliskan Hello Dunia sebanyak 10 kali sebagai berikut:</p> <pre> program while_do; </pre>

KEGIATAN	DISKRIPSI KEGIATAN										
	<pre> (*File : whiledo.pas*) (*Menggunakan fungsi loop while do untuk menulis hello dunia 10 kali*) uses crt; var i: integer; begin clrscr; i:= 0; while i < 10 do (* selama i<10 masuk loop *) begin writeln('Hello Dunia'); i:= i + 1; end; (* sampai di sini kembali ke WHILE *) (* untuk mengevaluasi kondisi *) end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada dalam program program P-AP-17, dengan kesimpulan sebagai berikut :</p> <table> <tr> <th>KODE</th><th>MAKNA</th></tr> <tr> <td>i:= 0;</td><td>Pada awal program, dibuat variabel i yang berfungsi sebagai variabel counter. Sebelum perulangan, kita memberikan nilai 0 untuk i. Setelah itu kita masuk ke perulangan WHILE DO.</td></tr> <tr> <td>while i < 10 do</td><td>Baris program while i < 10 do adalah awal dari perulangan. Inilah kondisi atau syarat yang harus dipenuhi supaya perulangan bisa diproses. Ketika kode program jalan pertama kali, nilai variabel i adalah 0, artinya kondisi i < 10 menghasilkan nilai TRUE, karena tentu saja 0 kurang dari 10.</td></tr> <tr> <td>writeln('Hello Dunia');</td><td>Karena syarat di penuhi, blok begin hingga end segera di eksekusi. Baris pertama adalah writeln ('Hello Dunia')</td></tr> <tr> <td>i:= i + 1; end;</td><td>Baris berikutnya adalah i:= i + 1. Bagian ini dikenal juga sebagai increment, artinya kita ingin menambah nilai variabel counter i sebanyak 1 angka. Ini dilakukan supaya bisa mengubah kondisi i < 10 yang terdapat di awal perulangan. Jika nilai i tidak pernah diubah, perulangan tidak akan pernah berhenti (infinity loop, looping).</td></tr> </table>	KODE	MAKNA	i:= 0;	Pada awal program, dibuat variabel i yang berfungsi sebagai variabel counter . Sebelum perulangan, kita memberikan nilai 0 untuk i. Setelah itu kita masuk ke perulangan WHILE DO .	while i < 10 do	Baris program while i < 10 do adalah awal dari perulangan. Inilah kondisi atau syarat yang harus dipenuhi supaya perulangan bisa diproses. Ketika kode program jalan pertama kali, nilai variabel i adalah 0, artinya kondisi i < 10 menghasilkan nilai TRUE , karena tentu saja 0 kurang dari 10.	writeln('Hello Dunia');	Karena syarat di penuhi, blok begin hingga end segera di eksekusi. Baris pertama adalah writeln ('Hello Dunia')	i:= i + 1; end;	Baris berikutnya adalah i:= i + 1. Bagian ini dikenal juga sebagai increment , artinya kita ingin menambah nilai variabel counter i sebanyak 1 angka. Ini dilakukan supaya bisa mengubah kondisi i < 10 yang terdapat di awal perulangan. Jika nilai i tidak pernah diubah, perulangan tidak akan pernah berhenti (infinity loop, looping).
KODE	MAKNA										
i:= 0;	Pada awal program, dibuat variabel i yang berfungsi sebagai variabel counter . Sebelum perulangan, kita memberikan nilai 0 untuk i. Setelah itu kita masuk ke perulangan WHILE DO .										
while i < 10 do	Baris program while i < 10 do adalah awal dari perulangan. Inilah kondisi atau syarat yang harus dipenuhi supaya perulangan bisa diproses. Ketika kode program jalan pertama kali, nilai variabel i adalah 0, artinya kondisi i < 10 menghasilkan nilai TRUE , karena tentu saja 0 kurang dari 10.										
writeln('Hello Dunia');	Karena syarat di penuhi, blok begin hingga end segera di eksekusi. Baris pertama adalah writeln ('Hello Dunia')										
i:= i + 1; end;	Baris berikutnya adalah i:= i + 1. Bagian ini dikenal juga sebagai increment , artinya kita ingin menambah nilai variabel counter i sebanyak 1 angka. Ini dilakukan supaya bisa mengubah kondisi i < 10 yang terdapat di awal perulangan. Jika nilai i tidak pernah diubah, perulangan tidak akan pernah berhenti (infinity loop, looping).										

KEGIATAN	DISKRIPSI KEGIATAN										
	<p>Setelah selesai guru meminta siswa menjelaskan hasilnya di depan kelas sebagai perwakilan, kemudian guru bertanya kepada siswa lain, terutama bagi yang belum bisa menyelesaikan, tentang kendala apa yang membuat mereka tidak bisa menyelesaikan kemudian memberi penjelasannya.</p> <p>Latihan loop-4 : REPEAT UNTIL</p> <p>Setelah selesai membahas perulangan dengan WHILE DO maka akan dibahas model perulangan yang terakhir yaitu menggunakan REPEAT UNTIL, guru mengajak siswa untuk membuat program P-AP-18 berikut :</p> <pre> program repeat_until; (File : repeatuntil.pas) (* menulis deret 5,10,15,...100 *) var i: integer; begin i:= 5; writeln('Berikut deret untuk deret dengan pertambahan 5: '); repeat begin write(i, ' '); i:= i + 5; end; until i > 100; readln; end. </pre> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada dalam program P-AP-16, dengan kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td>i:= 5;</td><td>Inisialisasi nilai variabel counter yaitu I dengan nilai awal 5.</td></tr> <tr> <td>REPEAT</td><td>Masuk ke perulangan REPEAT UNTIL</td></tr> <tr> <td>Begin write(i, ' '); i:= i + 5; end;</td><td>Badan pengulangan Menulis nilai I (yaitu 5,10,... 100) i:= i + 5 menambah nilai I dengan 5.</td></tr> <tr> <td>until i > 100;</td><td>Perintah ini menghentikan loop ketika i > 100</td></tr> </tbody> </table> <p>Latihan Loop-05 : BINTANG</p>	KODE PROGRAM	MAKNA	i:= 5;	Inisialisasi nilai variabel counter yaitu I dengan nilai awal 5.	REPEAT	Masuk ke perulangan REPEAT UNTIL	Begin write(i, ' '); i:= i + 5; end;	Badan pengulangan Menulis nilai I (yaitu 5,10,... 100) i:= i + 5 menambah nilai I dengan 5.	until i > 100;	Perintah ini menghentikan loop ketika i > 100
KODE PROGRAM	MAKNA										
i:= 5;	Inisialisasi nilai variabel counter yaitu I dengan nilai awal 5.										
REPEAT	Masuk ke perulangan REPEAT UNTIL										
Begin write(i, ' '); i:= i + 5; end;	Badan pengulangan Menulis nilai I (yaitu 5,10,... 100) i:= i + 5 menambah nilai I dengan 5.										
until i > 100;	Perintah ini menghentikan loop ketika i > 100										

KEGIATAN	DISKRIPSI KEGIATAN
	<p>Buatlah koding program P-AP-19 berikut :</p> <pre> Program Menampilkan_Bintang; (*File : bintang.pas*) (*Menampilkan logo bintang *) Var i,j,n:integer; Begin clrscr; Writeln('Program Menampilkan Bintang Bentuk Segitiga'); writeln('Dengan REPEAT-UNTIL'); Writeln('-----'); Writeln; Write('Berapa Jumlah Bitang: ');readln(n); i:=1; repeat for j:=1 to i do write('*'); writeln; i:=i+1; until (i>n); end. </pre> <p>Soal :</p> <ol style="list-style-type: none"> 1. Tuliskan output dari program P-AP-19 diatas ? 2. Program cetak bintang dengan kode program P-AP-17 menggunakan perulangan REPEAT UNTIL. <ol style="list-style-type: none"> a. Buatlah kode program yang menghasilkan output yang sama, namun menggunakan WHILE DO loop b. Buatlah kode program yang menghasilkan output yang sama, namun menggunakan FOR TO loop c. Buatlah kode program yang menghasilkan output yang sama, namun menggunakan FOR DOWNTO loop <p>Latihan Loop-06 : Problem Solving</p> <p>Buatlah sebuah program P-AP-18 tentang lagu anak ayam, sehingga tampilannya muncul seperti berikut :</p> 

KEGIATAN	DISKRIPSI KEGIATAN
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisi pada form yang sudah dibuat

7.6.5 Lembar Kerja Siswa

- (1) LKS berupa progcil – mengerjakan **Latihan Loop-05 : program BINTANG** di halaman 70
- (2) LKS berupa Problem Solving – mengerjakan **Latihan Loop-05 : Problem Solving** di halaman 71

7.6.6 Assesment

Rubrik Procil pada Lembar Kerja Siswa – (1)

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Kebenaran Program tampilan Bintang	Program dapat menampilkan bintang sesuai ketentuan	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Kesalahan sintaks.	Program dapat berjalan dengan menggunakan WHILE DO loop, FOR TO loop dan FOR DOWNT0 loop	Program dapat berjalan dengan menggunakan 2 dari 3 jenis loop yang telah ditentukan	Program dapat berjalan dengan menggunakan 1 dari 3 jenis loop yang telah ditentukan	Program tidak dapat berjalan dengan menggunakan 3 jenis loop yang telah ditentukan
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengkoreksi secara mandiri	Siswa sese kali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

Rublik Problem Solving pada Lembar Kerja Siswa – (2)

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Mampu membuat notasi algoritma	Mampu membuat rancangan notasi algoritma berupa flowchart dan pseudocode	Mampu membuat rancangan notasi algoritma berupa flowchart saja	Mampu membuat rancangan notasi algoritma berupa pseudocode saja	Tidak mampu membuat rancangan notasi algoritma berupa flowchart dan pseudocode

Design input-output-proses	Siswa mampu membuat 3 elemen disain input, proses dan output untuk persoalan yang diberikan	Siswa mampu membuat satu dari 2 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat satu dari 3 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat disain input, proses dan output untuk persoalan yang diberikan
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program menggunakan loop	Program dapat berjalan dengan menggunakan fungsi loop	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sese kali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

7.6.7 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami dan dapat menerapkan perintah perulangan?	
Apakah siswa mampu mendeteksi kesalahan program yang tidak bisa berjalan?	
Apakah senang belajar materi perulangan dalam bahasa pemrograman pascal?	
Apakah terdapat kesulitan dalam belajar dalam materi perulangan dalam bahasa pemrograman pascal?	

7.6.8 Contoh Soal Ulangan

Perhatikan potongan program berikut :

```

program PRIREP;
var
    N : integer;
    I : integer;

begin (* Program *) write ( 'Nilai N= '); readln (N);
    i := 1 ;
    writeln ( 'Print i dengan REPEAT: ');

```

```

repeat
    writeln (i); i := 1;
until (i > N);
end.

```

Soal :

1. Apakah program diatas dapat berjalan dengan baik ?
2. Apabila nilai N diberikan 5 tuliskan output yang dihasilkan program tersebut.

7.7 Aktivitas-8 Array

Kode Aktivitas : K10-AP-P6-A7-Array

Pada aktivitas ini, siswa akan belajar tentang tipe data array, yang perlu dibahas karena merupakan salah satu struktur data yang sudah disediakan oleh bahasa pemrograman pascal untuk memproses sekumpulan nilai. Tanpa array, sulit sekali menulis sebuah program yang ringkas dan efisien untuk memproses koleksi (himpunan) data. Misalnya jika dibutuhkan untuk memproses data nomor siswa (integer) untuk 100 siswa yang semua datanya harus berada dalam memori, maka kita harus membuat deklarasi 100 nama variabel karena setiap variabel hanya dapat menampung 1 nilai saja. Apalagi kalau datanya ribuan.

Pertanyaannya, bagaimana membuat deklarasi sebuah nama yang dapat menampung 10, 30, 56, 100, 1000 atau N buah nilai ?

Keseluruhan aktivitas diperkirakan akan memerlukan 1 pertemuan selama 3 JP.

Materi yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pembuatan program dengan berbagai jenis materi diatas menggunakan free pascal atau ideone.com.

7.7.1 Pertanyaan Pemantik

Pernahkah kalian mengisi Daftar Hadir ? Satu baris daftar hadir adalah untuk kehadiran 1 siswa. Kalau ada 30 siswa dalam rombongan itu, ada berapa baris dalam daftar hadir ?

7.7.2 Kata Kunci

Array, elemen, indeks

7.7.3 Konsep

A. Pengertian Tipe Data Array

Array adalah suatu tipe atau struktur data yang dapat menyimpan sekumpulan elemen dengan tipe yang sama. Setiap elemen dapat di akses secara langsung melalui indeksnya. Array T yang berukuran N akan berisi (menyimpan nilai) sekumpulan elemen $T_1, T_2, T_3, \dots, T_N$ dengan 1, 2, 3, ..., N adalah **indeks**, yang tipe datanya harus type ordinal. Jadi satu nama T akan dipakai bersama oleh N nilai integer. Perhatikan namanya tetap T tetapi ada tambahan indeks.

$T_1, T_2, T_3, \dots, T_N$ berbeda dengan $T_1, T_2, T_3, \dots, T_N$.

Daripada membuat 10 variabel yang terdiri dari nama1, nama2, nama3, dst, akan lebih efisien jika variabel nama ini dijadikan array. Apalagi jika indeks nya sampai ribuan.

Array mewakili data berstruktur linier, biasa juga disebut sebagai Tabel atau Vektor. Array seringkali digambarkan sebagai kotak-kotak (lokasi memori) berindeks. Sebuah array Tab dengan indeks 1 s.d. 6 yang “isi” (nilai yang disimpan) adalah {10,20,30, 4,0,5} dapat diilustrasikan sebagai berikut

Tab

10	20	30	4	0	5
1	2	3	4	5	6

Kita juga dapat meng gambarkannya secara vertikal dan bukan horizontal. Contoh berikut ini menunjukkan sebuah array yang semua elemennya bernilai 0

1	0
2	0
3	0
4	0
5	0
6	0

Perhatikan yang dimasukkan “kotak” hanya nilai yang disimpan

Saat memprogram, kita perlu memikirkan apakah memang perlu memakai array sebab array membutuhkan memori yang langsung dialokasi saat variabel bertipe array dideklarasikan. Kita perlu memesan array jika semua data yang diolah perlu disimpan untuk komputasi, dan saat melakukan deklarasi juga memesan array sesuai kapasitas yang dibutuhkan saja. Contohnya : 5 orang yang akan minum air dari sebuah teko, dan hanya tersedia 1 gelas, maka dapat meminum air satu per satu dengan memakai gelas secara bergiliran. Namun, jika ingin berfoto di mana akan ditunjukkan kebersamaan bahwa kelima orang tersebut minum secara bersama-sama, maka diperlukan 5 gelas.

Contoh kasus sebuah program yang membutuhkan array atau tidak membutuhkan array:

- Saat membaca sekumpulan nilai untuk menghitung nilai rata-ratanya, kita tidak perlu menyimpan data yang dibaca dalam array, karena setiap kali data dibaca, dapat langsung dijumlahkan sambil mencacah. Di akhir pengulangan, kita dapat menghitung rata-rata. Contoh: jika yang dibaca adalah 1,5,1,2,6,3 maka nilai rata-rata adalah $18/6=3$.
Sebuah program yang membaca huruf demi huruf dan menghasilkan menuliskan “terbalik”, kita perlu menyimpan setiap huruf yang dibaca ke dalam array karena huruf pertama yang dicetak adalah huruf yang terakhir dibaca. Tanpa array, maka huruf yang sudah dibaca akan tertimpa oleh huruf berikutnya. Contoh : jika deretan karakter yang dibaca adalah ‘A’, ‘N’, ‘A’, ‘K’, ‘ ‘, ‘S’, ‘M’, ‘A’ maka yang harus ditulis adalah ‘A’, ‘M’, ‘S’, ‘ ‘, ‘K’, ‘A’, ‘N’, ‘A’.

Dapatkah siswa membuat program tersebut di atas ? Pikirkan!

B. Cara Penggunaan Tipe Data Array Pascal

Untuk membuat tipe data array dalam program Pascal, kita harus menentukan rentang nilai indeksinya, yang akan menentukan ukuran atau kapasitasnya, yaitu banyaknya elemen yang dapat ditampung dalam array tersebut. Elemen adalah sebutan untuk 'anggota' / isi dari array. Sebagai contoh, untuk membuat sebuah array bernama NILAI yang dapat menampung 10 elemen bertipe integer dengan rentang nilai indeks 0 s.d. 9 maka deklarasi variabelnya adalah sebagai berikut: (deklarasi-A)

```
var
    NILAI: array[0..9] of integer;
```

Tentu, kita dapat menentukan array dengan kapasitas 10 dengan indeks bukan 0..9 tetapi 1..10, sehingga deklarasinya menjadi sebagai berikut : (deklarasi-B)

```
var
    NILAI: array[1..10] of integer;
```

Dalam bahasa Pascal, kita bebas menentukan rentang nilai indeksinya. Rentang nilai indeks tersebut akan menentukan kapasitas atau banyaknya elemen yang dapat disimpan.

Cara mengakses element tersebut adalah melalui indeks. Indeks adalah alamat memori elemen di dalam sebuah array. Sebagai contoh, untuk mengakses elemen ke – 2, kita bisa menulis: NILAI[2]. Untuk mengakses element ke-6, bisa menggunakan: Nilai[6]. Kita hanya dapat mengakses elemen NILAI dengan indeks sesuai dengan rentang yang dideklarasikan. Misalnya, jika kita mengakses NILAI[10] padahal deklarasi yang dipakai adalah cara pertama yaitu dengan indeks 0..9, maka akan terjadi error. Akses dengan NILAI[20] untuk kedua deklarasi akan salah sebab 20 di luar rentang nilai indeks saat array NILAI dideklarasikan. Pada Freepascal, array juga dapat diinisialisasi nilai elemennya pada saat deklarasi seperti halnya variabel biasa. Berikut ini contoh untuk menginisialisasi array pada saat deklarasi

```
VAR
    T:array[1..3] of integer= (1,2,3);
    TC: array [1..5] of char = ('A', '0', '@');
    TabBit : array [1..4] of boolean = (true, true, false,true);
    TabNama : array [1..4] of string = ('Ani', 'Ana', 'Bob', 'Siti')
```

Perhatikan bahwa banyaknya nilai yang diisikan harus sama dengan kapasitas array yang kita pesan, dan type nilainya harus sama dengan type elemen array.

Array 1 dimensi adalah sebuah array yang indeksinya hanya 1 sumbu seperti contoh di atas, seperti nilai[0], nilai[1] dan nilai[2]. Bahasa Pascal memungkinkan kita membuat array 2 dimensi, 3 dimensi atau lebih. Namun, yang dibahas pada pelajaran ini hanya array satu dimensi yang biasa disebut Tabel dengan elemen T[i], dengan i sesuai rentang indeks yang didefinisikan

Referensi : <https://wiki.freepascal.org/Array>

7.7.4 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN						
Pendahuluan	Pada awal pembelajaran, guru menjelaskan itu array, array sangat dibutuhkan ketika kita menyelesaikan kasus-kasus dengan bentuk 3 dimensi seperti membuat grafik.						
Inti	<p>Latihan 01- array</p> <p>Bacalah dengan cermat contoh Program P-AP-20 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre> program Contoh_array; (*File: array.pas*) (* deklarasi dan mengisi tabel integer dengan 5 elemen*) (* yang indeks nya mulai daari 0 s.d. 4 *) var nilai: array [0..4] of integer; begin clrscr; nilai[0]:=80; nilai[1]:=67; nilai[2]:=56; nilai[3]:=90; nilai[4]:=78; writeln('Nilai indeks ke 0 : ',nilai[0]); writeln('Nilai indeks ke 1 : ',nilai[1]); writeln('Nilai indeks ke 2 : ',nilai[2]); writeln('Nilai indeks ke 3 : ',nilai[3]); writeln('Nilai indeks ke 4 : ',nilai[4]); end. </pre> <p>Saat program selesai, data kita “hilang” tidak tersimpan. Kenapa demikian? karena array adalah memori internal program. Saat program selesai dijalankan, semua data yang disimpan dalam memori akan hilang. Supaya tidak hilang, harus disimpan dalam memori eksternal, misalnya dalam sebuah file. Pada pemrograman dasar ini, file tidak dicakup.</p> <p>Kemudian guru bersama siswa mendiskusikan tentang perintah-perintah yang ada dalam program P-AP-20 di atas, dengan kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td> <pre> Var nilai: array [0..4] of integer; </pre> </td><td>Deklarasi sebuah array, bernama <i>nilai</i> yang indeks nya mulai dari 0 s.d. 4 dan setiap elemennya bernilai integer.</td></tr> <tr> <td> <pre> nilai[0]:=80; nilai[1]:=67; nilai[2]:=56; nilai[3]:=90; </pre> </td><td> <p>Memberikan nilai pada elemen nilai indek ke 0 adalah 80</p> <p>Memberikan nilai pada elemen nilai indek ke 1 adalah 67</p> </td></tr> </tbody> </table>	KODE PROGRAM	MAKNA	<pre> Var nilai: array [0..4] of integer; </pre>	Deklarasi sebuah array, bernama <i>nilai</i> yang indeks nya mulai dari 0 s.d. 4 dan setiap elemennya bernilai integer.	<pre> nilai[0]:=80; nilai[1]:=67; nilai[2]:=56; nilai[3]:=90; </pre>	<p>Memberikan nilai pada elemen nilai indek ke 0 adalah 80</p> <p>Memberikan nilai pada elemen nilai indek ke 1 adalah 67</p>
KODE PROGRAM	MAKNA						
<pre> Var nilai: array [0..4] of integer; </pre>	Deklarasi sebuah array, bernama <i>nilai</i> yang indeks nya mulai dari 0 s.d. 4 dan setiap elemennya bernilai integer.						
<pre> nilai[0]:=80; nilai[1]:=67; nilai[2]:=56; nilai[3]:=90; </pre>	<p>Memberikan nilai pada elemen nilai indek ke 0 adalah 80</p> <p>Memberikan nilai pada elemen nilai indek ke 1 adalah 67</p>						

KEGIATAN	DISKRIPSI KEGIATAN	
	<code>nilai[4]:=78;</code>	Memberikan nilai pada elemen nilai indek ke 2 adalah 56 Memberikan nilai pada elemen nilai indek ke 3 adalah 90 Memberikan nilai pada elemen nilai indek ke 4 adalah 78
	<code>writeln('Nilai indeks ke 0 : ',nilai[0]);</code>	Menuliskan teks 'Nilai indek ke 0 : 80'
	<code>writeln('Nilai indeks ke 1 : ',nilai[1]);</code>	Menuliskan teks 'Nilai indek ke 1 : 67'
	<code>writeln('Nilai indeks ke 2 : ',nilai[2]);</code>	Menuliskan teks 'Nilai indek ke 2 : 56'
	<code>writeln('Nilai indeks ke 3 : ',nilai[3]);</code>	Menuliskan teks 'Nilai indek ke 3 : 90'
	<code>writeln('Nilai indeks ke 4 : ',nilai[4]);</code>	Menuliskan teks 'Nilai indek ke 4 : 78'

Latihan 02 – Membaca dan Menulis isi Array

Untuk Latihan 01 – array dengan kode program P-AP-20 di atas mudah dituliskan karena array hanya mempunyai 5 elemen. Jika elemennya ada 1000 ? maka cara mengisi dengan operator *assignment* tentunya tidak efisien. Selain itu, jika data berubah, maka kita harus mengubah program. Pada latihan berikut ini, kita akan mengubah pengisian array dengan membaca data yang diketikkan dari papan kunci, kemudian menuliskannya. Bacalah dengan cermat contoh Program P-AP-21 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :

```

program TABEL;
(* File : TABEL.PAS *)
(* latihan array : mengisi dg assignment, menulis *)
VAR
(* Kamus *)
i : integer;
tab : array [1..1000] of integer;
begin
    (* Program *)
    writeln ( 'Isi dan print tabel: ');
    (* isi dengan assignment *)
    for i := 1 to N do
    begin
        tab [i] := i;
    end;
end;

```

KEGIATAN	DISKRIPSI KEGIATAN												
	<pre> (* telusuri array, print *) for i := 1 to N do begin writeln ('i=',i, 'tab[i]=' , tab[i]); end; end. </pre> <p>Dari kode program P-AP-21 diatas guru dan siswa berdiskusi untuk memahami kode program dan maknanya, sehingga dapat ditarik kesimpulan sebagai berikut :</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td>Var i:integer;</td><td>Deklarasi variabel yang menjadi indeks tab</td></tr> <tr> <td>tab : array [1..1000] of integer;</td><td>Deklarasi sebuah array, bernama tab yang indeksny mulai dari 1 s.d. 1000 dan setiap elemennya bernilai integer.</td></tr> <tr> <td>writeln ('Isi dan print tabel: ');</td><td>Menuliskan teks 'Isi dan print tabel: ' di awal, hanya supaya kita mengetahui bahwa program dapat berjalan</td></tr> <tr> <td>for i := 1 to N do begin tab [i] := i; end;</td><td>Perintah untuk mengisi nilai tab[i] dengan i artinya isi dari tabel sama dengan indeksny. Hasilnya adalah: tab[1] berisi 1, tab[2] berisi 2, ... tab[1000] berisi 1000.</td></tr> <tr> <td>for i := 1 to N do begin writeln ('i=',i, 'tab[i]=' , tab[i]); end;</td><td>Perintah untuk menuliskan setiap nilai tab[i] sehingga hasilnya : 1 ... 1000</td></tr> </tbody> </table> <p>Aktivitas Kreatif siswa : Perhatikan bahwa program mengandung dua loop FOR. Apakah bisa dilakukan dengan hanya menggunakan satu <i>loop</i> saja ? jelaskan Jika mungkin, tuliskan program di atas hanya menggunakan satu <i>loop</i> saja. Apa akibatnya ?</p> <p>Latihan 03 – Maksimum Setelah data disimpan dalam sebuah Array, proses yang paling sering dilakukan adalah mencari nilai paling besar (maksimum) atau paling kecil (minimum) yang disimpan.</p> <p>Berikut ini adalah contoh menentukan nilai paling besar dari sekumpulan data yang disimpan dalam array. Program P-AP-22 ini hanya contoh, karena</p>	KODE PROGRAM	MAKNA	Var i:integer;	Deklarasi variabel yang menjadi indeks tab	tab : array [1..1000] of integer;	Deklarasi sebuah array, bernama tab yang indeksny mulai dari 1 s.d. 1000 dan setiap elemennya bernilai integer.	writeln ('Isi dan print tabel: ');	Menuliskan teks 'Isi dan print tabel: ' di awal, hanya supaya kita mengetahui bahwa program dapat berjalan	for i := 1 to N do begin tab [i] := i; end;	Perintah untuk mengisi nilai tab[i] dengan i artinya isi dari tabel sama dengan indeksny. Hasilnya adalah: tab[1] berisi 1, tab[2] berisi 2, ... tab[1000] berisi 1000.	for i := 1 to N do begin writeln ('i=',i, 'tab[i]=' , tab[i]); end;	Perintah untuk menuliskan setiap nilai tab[i] sehingga hasilnya : 1 ... 1000
KODE PROGRAM	MAKNA												
Var i:integer;	Deklarasi variabel yang menjadi indeks tab												
tab : array [1..1000] of integer;	Deklarasi sebuah array, bernama tab yang indeksny mulai dari 1 s.d. 1000 dan setiap elemennya bernilai integer.												
writeln ('Isi dan print tabel: ');	Menuliskan teks 'Isi dan print tabel: ' di awal, hanya supaya kita mengetahui bahwa program dapat berjalan												
for i := 1 to N do begin tab [i] := i; end;	Perintah untuk mengisi nilai tab[i] dengan i artinya isi dari tabel sama dengan indeksny. Hasilnya adalah: tab[1] berisi 1, tab[2] berisi 2, ... tab[1000] berisi 1000.												
for i := 1 to N do begin writeln ('i=',i, 'tab[i]=' , tab[i]); end;	Perintah untuk menuliskan setiap nilai tab[i] sehingga hasilnya : 1 ... 1000												

KEGIATAN	DISKRIPSI KEGIATAN
	<p>kalau hanya untuk membaca data dan menentukan nilai maksimum dari sekumpulan data yang dibaca, tidak diperlukan array!</p> <pre> program maksimum; (*File : maxarray.pas*) (* deklarasi dan inisiasi sebuah array, menuliskan nilai yang maksimum *) VAR T: array [1..10] of integer =(2,5,1,80,100,123,5,55,23,-3); i: integer; Max: integer; // nilai maksimum yang dicari begin Max:=T[1]; // elemen pertama dianggap yang maksimum // selanjutnya, bandingkan dengan elemen lainnya for i:= 2 TO 10 do begin if (T[i] > Max) then begin Max:= T[i]; end; end; writeln ('Nilai paling besar adalah : ', Max); end. </pre> <p><u>Aktivitas kreatif siswa :</u></p> <p>setelah memahami program P-AP-22 di atas, andaikata data tidak diinisialisasi, tetapi dibaca dengan perintah readln, tuliskan kodenya tanpa menggunakan array.</p> <p>Latihan 04 – Tentukan minimum</p> <p>Ubahlah sehingga program P-AP-22 untuk menentukan nilai paling besar sekarang menghasilkan nilai terkecil. Kode proram yang digunakan dalam latihan minimum ini adalah P-AP-23.</p> <p>(Siswa tidak diberikan kodenya. Untuk guru : hanya mengubah Max menjadi Min, dan mengganti kode baris 11 menjadi T[i]< Min)</p> <p><u>Aktivitas kreatif siswa :</u></p> <p>Andaikata program P-AP-23 sudah diubah untuk mencari minimum dan data elemen array dibaca. Sebetulnya, apakah memang perlu disimpan dalam array? jika bisa tanpa array, tuliskanlah programnya</p> <p>Latihan 05 - SearchSelain nilai paling besar dan paling kecil, seringkali kita perlu mencari apakah elemen yang disimpan dalam sebuah array ada yang bernilai X?</p> <p>Ada banyak versi kode program untuk mencari apakah sebuah array mengandung sebuah nilai X, dengan memeriksa nilainya satu persatu (secara sekuensial). Berikut ini adalah salah satu kode paling <i>robust</i> dan efisien untuk mencari nilai, dengan menggunakan loop WHILE.</p>

KEGIATAN	DISKRIPSI KEGIATAN										
	<p>Bacalah dengan cermat contoh Program P-AP-24 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre> program SearchX; (*File:searcharray.pas*) (*mencari apakah nilai yang diketikkan ada pada array *) (*yang sudah diisi nilainya saat deklarasi *) var T: array [1..10] of integer = (2,5,1,80,100,123,5,55,23,-3); i: integer; X: integer; // nilai yang dicari, dibaca dari papan ketikan begin readln (X); // elemen pertama dianggap yang maksimum // selanjutnya, bandingkan dengan elemen lainnya while (i < 10) and (T[i] <> X) do begin i := i+ 1; end; (* i=10 OR T[i]=X *) if (T[i]=X) then begin writeln ('Nilai',X,'ketemu pada indeks ke ', i); end else begin writeln ('Tidak ada elemen bernilai ', X); end end. </pre> <p>Dari program P-AP-24 guru dan siswa berdiskusi untuk memahami kode program dan maknanya, sehingga dapat ditarik kesimpulan sebagai berikut:</p> <table border="1"> <thead> <tr> <th>KODE PROGRAM</th><th>MAKNA</th></tr> </thead> <tbody> <tr> <td>T: array [1..10] of integer = = (2,5,1,80,100,123,5,55,23,-3);</td><td>Deklarasi dan inisialisasi array T, sehingga nilainya : T[1]= 2, T[2]= 5, T[3]=1, T[4]=80, T[5]=100, T[6]=123, T[7]=5, T[8]=55, T[9]=23, T[10]=-3</td></tr> <tr> <td>Begin readln (X);</td><td>Membaca nilai X yang akan dicari</td></tr> <tr> <td>while (i < 10) and (T[i] <> X) do begin i := i+ 1; end;</td><td>Proses membandingkan setiap elemen array T[i] dengan X. Loop dihentikan jika nilai i sudah mencapai 10, atau T[i]=X (ketemu).</td></tr> <tr> <td>if (T[i]=X) then begin</td><td>Saat loop berhenti, ada dua kemungkinan: <ul style="list-style-type: none"> i sudah mencapai 10 </td></tr> </tbody> </table>	KODE PROGRAM	MAKNA	T: array [1..10] of integer = = (2,5,1,80,100,123,5,55,23,-3);	Deklarasi dan inisialisasi array T, sehingga nilainya : T[1]= 2, T[2]= 5, T[3]=1, T[4]=80, T[5]=100, T[6]=123, T[7]=5, T[8]=55, T[9]=23, T[10]=-3	Begin readln (X);	Membaca nilai X yang akan dicari	while (i < 10) and (T[i] <> X) do begin i := i+ 1; end;	Proses membandingkan setiap elemen array T[i] dengan X. Loop dihentikan jika nilai i sudah mencapai 10, atau T[i]=X (ketemu).	if (T[i]=X) then begin	Saat loop berhenti, ada dua kemungkinan: <ul style="list-style-type: none"> i sudah mencapai 10
KODE PROGRAM	MAKNA										
T: array [1..10] of integer = = (2,5,1,80,100,123,5,55,23,-3);	Deklarasi dan inisialisasi array T, sehingga nilainya : T[1]= 2, T[2]= 5, T[3]=1, T[4]=80, T[5]=100, T[6]=123, T[7]=5, T[8]=55, T[9]=23, T[10]=-3										
Begin readln (X);	Membaca nilai X yang akan dicari										
while (i < 10) and (T[i] <> X) do begin i := i+ 1; end;	Proses membandingkan setiap elemen array T[i] dengan X. Loop dihentikan jika nilai i sudah mencapai 10, atau T[i]=X (ketemu).										
if (T[i]=X) then begin	Saat loop berhenti, ada dua kemungkinan: <ul style="list-style-type: none"> i sudah mencapai 10 										

KEGIATAN	DISKRIPSI KEGIATAN	
	<pre> writeln ('Nilai', X, ' ketemu pada indeks ke ', i); end else begin writeln ('Tidak ada elemen bernilai ', X); end. </pre>	<ul style="list-style-type: none"> • T[i]=X (ketemu) <p>Pada kondisi i sudah mencapai 10, elemen T[10] belum diperiksa. Maka ditulis sebuah pernyataan kondisional untuk memeriksa apakah sama atatau tidak sama, dan menuliskan. Perhatikan bahwa pada program ini, elemen T[10] diperiksa di luar loop</p>
	Setelah selesai pembahasan tentang materi array selesai, siswa diminta untuk mengerjakan soal pada Lembar Kerja siswa.	
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisi pada form yang sudah dibuat	

7.7.5 Lembar Kerja Siswa

Menegerjakan **Latihan 04 – Tentukan nilai minimum** menggunakan LKS Procil dengan kode P-AP-23

7.7.6 Assesment

Digunakan untuk menilaia LKS

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program mencari nilai minimum pada array	Program dapat mencari nilai minimum pada array	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengkoreksi secara mandiri	Siswa sesekali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

7.7.7 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami dan dapat menerapkan array pada sebuah kasus?	
Apakah siswa mampu mendeteksi kesalahan program yang tidak bisa berjalan dengan menggunakan array?	
Apakah senang belajar materi array dalam bahasa pemrograman pascal?	
Apakah terdapat kesulitan dalam belajar dalam materi array dalam bahasa pemrograman pascal?	

7.7.8 Contoh Soal Ulangan

Buatlah program berikut :

```

Program Deklarasi_Array_Beragam;
Var
    NPM    : array[1..20] of string[10];
    Nama   : array[1..20] of string[25];
    Nilai  : array[1..20] of real;
    Umur   : array[1..20] of byte;
    banyak,i    : integer;
begin
    write('Isi berapa data array yang diperlukan :');
    readln(banyak);
    for i := 1 to banyak Do
    begin
        write('NPM =');Readln(NPM[i]);
        write('Nama =');readln(Nama[i]);
        write('Nilai=');readln(Nilai[i]);
        write('Umur =');readln(Umur[i]);
    end;
end.

```

Dari program diatas tentukan :

1. Tampilan output apa yang muncul
2. Jelaskan setiap baris programnya

7.8 Aktivitas-7 Subprogram

Kode Aktivitas : K10-AP-P7-A8-Subprogram

Pada aktivitas ini, siswa akan belajar tentang subprogram, yaitu *function* dan *procedure* dalam bahasa pascal. Keseluruhan aktivitas diperkirakan akan memerlukan 1 pertemuan selama 3 JP.

Materi yang akan dipelajari dalam aktivitas ini akan ditutup dengan latihan pembuatan program dengan berbagai jenis materi diatas menggunakan kompiler free pascal atau ideone.com.

7.8.1 Pertanyaan Pemantik

Pernahkan kalian melakukan komputasi untuk menentukan sebuah nilai terbesar dalam sebuah deretan angka?

7.8.2 Kata Kunci

Subprogram, function, procedure

7.8.3 Konsep Terkait Aktivitas

Subprogram adalah potongan program yang melakukan tugas tertentu, yang akan “dipanggil” oleh program utama atau subprogram yang lain. Sebuah subprogram dapat dipanggil oleh program, yang disebut program pemanggil. Sub program merupakan bagian dari program, yang dapat dituliskan dalam 1 file bersama dengan program utamanya, atau dapat dituliskan dalam 1 file kemudian diasembly bersama program utama. Pada aktivitas ini, siswa hanya belajar program yang ditulis dalam satu file saja.

Bahasa Pascal adalah bahasa berstruktur blok yang bersarang. Sebuah subprogram dituliskan di “dalam” program utama. Sebuah subprogram dapat mengandung subprogram yang lain.

Dalam bahasa Pascal, subprogram dapat ditulis sebagai fungsi, atau prosedur.

Tujuan dibuatnya prosedur atau fungsi adalah untuk memudahkan proses coding, karena setiap tugas atau fungsi nantinya dapat di kelompokkan ke dalam satu blok kode yang diberi nama, dan dapat “dipanggil” sehingga proses yang kelihatannya rumit dapat didekomposisi menjadi bagian-bagian yang lebih sederhana. Dalam teks bahasa sehari-hari, sub program ibarat memisahkan teks dalam lampiran, yang hanya dirujuk dalam teks utama, untuk mempermudah pemahaman dalam membaca teks. Selain itu, jika komputasi yang dilakukan dibutuhkan lebih dari satu kali, maka teks program menjadi lebih ringkas.

Beberapa pengertian terkait subprogram:

- A. Subprogram : adalah potongan program, dalam bahasa Pascal dapat berupa fungsi (function) atau prosedur (procedure), yang dapat “**dipanggil**” oleh program utama atau fungsi/prosedur lain. Selanjutnya, istilah “subprogram” mencakup fungsi dan prosedur.
- B. Program utama atau subprogram dapat “**memanggil**” subprogram. Jadi, ada pengertian tentang program/subprogram “pemanggil” dan subprogram “yang dipanggil”. Dalam bahasa Pascal, program utama hanya dapat memanggil subprogram. Program utama tidak dapat dipanggil.
- C. Pada bahasa Pascal standar yang hanya terdiri dari satu file, subprogram harus didefinisikan di dalam sebuah program utama. Subprogram yang didefinisikan pada file lain membentuk unit-unit program terpisah, belum merupakan pembahasan pada modul ajar ini.
- D. Subprogram dapat mempunyai parameter atau tanpa parameter. Parameter memungkinkan subprogram untuk dieksekusi dengan nilai yang berbeda-beda. Mirip dengan parameter fungsi yang telah dikenal pada matematika. Misalnya dengan fungsi persamaan kuadrat $f(x) = x^2 + x + 3$ yang akan dapat dipakai menghitung nilai $f(x)$ untuk berbagai nilai x . Parameter x dari sebuah fungsi bernama “ f ” dituliskan pada definisi fungsinya yaitu $f(x)$

- E. Parameter sub program:
 - a. Parameter dapat diubah atau menjadi hasil komputasi, atau hanya sebagai input (tidak dapat diubah).
 - b. Nama parameter yang dituliskan dalam pendefinisian subprogram dapat berbeda dengan nama variabel yang dipakai saat subprogram dipanggil, tapi TYPE nya harus sama.
- F. Lingkup (*scope*) suatu variabel, konstanta :
 - a. Lingkup suatu nama variabel atau nama konstanta menentukan di mana nama tersebut dikenal, yaitu di dalam blok tempat nama tersebut didefinisikan, dan semua blok yang dilingkupinya (di dalamnya).
 - b. Suatu nama variabel atau nama konstanta berlaku untuk lingkup di mana nama variabel atau nama konstanta tersebut didefinisikan. Jika ada nama variabel yang sama, maka program akan mengambil nilai dari nama yang lingkupnya paling dalam. Akan dijelaskan lebih lanjut lewat contoh
 - c. Nama variabel yang sama dalam lingkup yang berbeda tidak akan menimbulkan masalah sebab akan dikenali sesuai lingkupnya.
 - d. Jika sebuah prosedur atau fungsi memakai nama yang tidak didefinisikan dalam definisinya, maka nilainya akan diambil dari blok dengan lingkup “terdekat”, yang terakhir adalah program utama.
 - e. Nama variabel yang didefinisikan pada program utama disebut variabel global. Sedangkan nama variabel yang didefinisikan pada subprogram disebut sebagai variabel lokal.
- G. Masa hidup (*life time*) suatu nama variabel atau nama konstanta
 - a. Masa hidup (masa berlaku) sebuah variabel adalah selama prosedur atau fungsi sedang dijalankan. Saat sebuah prosedur atau fungsi selesai dieksekusi (mencapai “end” terkait), semua nama variabel dan nama konstanta yang didefinisikan sudah tidak dikenal lagi karena memorinya sudah dibebaskan.
 - b. Dari penjelasan di atas, masa hidup variabel yang paling lama adalah nama variabel dan nama konstanta program utama, yang akan berakhir saat program utama berakhir atau mencapai “end.”. Variabel yang didefinisikan pada program utama disebut **variabel global**.

Bahasa Pascal adalah bahasa yang berstruktur blok. Contoh dari sebuah program utama dengan beberapa prosedur dan fungsi yang bersarang serta parameternya diberikan sebagai berikut. Contoh ini menunjukkan bahwa dalam sebuah program dapat mengandung fungsi atau prosedur. Selanjutnya, sebuah fungsi, dapat mengandung fungsi lain atau prosedur. Demikian pula bahwa sebuah prosedur juga dapat mengandung prosedur dan fungsi.

```

Program utama;
(* hanya contoh *)
  VAR i:integer;
    procedure P1 (k:integer);
      VAR
        i:integer;
        procedure P11;
          VAR i: integer;
          begin

```

```

end; (* P11 *)

function FP1(i:integer) : real;
begin
end;
begin (* P1*)
(* body Prosedur P1 *)
end;

Function F1 : integer;
procedure PF1;
begin
end; (* PF1 *)

function FF1 (x:real):integer;
begin
(* body fungsi FF1*)
end; (* FF1 *)
begin
(* body fungsi F1 *)
end; (* F1 *)
begin
writeln ('contoh block pascal ')
end.

```

A. Function

Function adalah sekelompok pernyataan untuk melakukan komputasi yang hasilnya akan dikirimkan ke program pemanggilnya. Fungsi dalam sebuah program diimplementasi sangat mirip dengan konsep “fungsi” dalam matematika: sebuah fungsi akan menerima parameter, melakukan komputasi, dan mengirimkan hasil komputasi ke program pemanggilnya. Dalam bahasa Pascal, hasil komputasi hanya perlu ditentukan TYPE, dan merupakan TYPE dari Fungsi

A.1. Pendefinisian Function

Pendefinisian fungsi sama halnya dengan prosedur, yang membedakan pada kata kunci menggunakan function sebelum nama fungsi di deklarasikan.

Perbedaan antara function dengan procedure adalah bahwa function mempunyai TYPE dari nilai hasil komputasinya. Tipe kembalian sebuah fungsi harus berupa tipe dasar. Pada akhir eksekusi, fungsi akan mengembalikan nilai sesuai dengan tipe fungsi tersebut.

Perlu diketahui setiap nama variabel/konstanta yang di deklarasikan di dalam fungsi hanya dapat digunakan dalam fungsi itu sendiri dan subprogram yang di dalam lingkupnya; tidak dikenal oleh subprogram yang melingkupinya atau program utama.

Berikut ini bentuk umum pendefinisian prosedur di pascal:

```
Program Nama_Program;
```

```

Function nama_fungsi(daftar_parameter): type;
Begin
    (* Badan Fungsi *)
    Nama_fungsi := ekspresi; //akhir eksekusi fungsi
End;
Begin
    (* Badan Program utama *)
End.

```

Perhatikan bahwa eksekusi fungsi diakhiri dengan menyimpan nilai hasil komputasi pada “nama” fungsi tersebut.

B. Procedure (prosedur)

Prosedur adalah potongan teks sebuah program dan dapat diaktifkan (“dipanggil”) dimanapun didalam program. Prosedur ibarat sebuah teks yang dirujuk untuk “ditempel” saat “dipanggil”. Prosedur ibarat sebuah lampiran yang ditulis di luar teks utama, yang diacu (“dipanggil”) untuk diaktifkan.

B.1. Pendefinisian Prosedur

Mendefinisikan prosedur artinya menuliskan nama prosedur, parameternya, dan menjabarkan deretan instruksi yang akan dilakukan

Berikut ini bentuk umum pendefinisian prosedur yang tidak mengandung parameter, dalam bahasa pascal:

```

Program Nama_Program;
Procedure Nama_Prosedur;
Begin
    (* Badan prosedur *)
End;
Begin
    (* Badan Program utama *)
End.

```

prosedur yang mengandung parameter yang tidak diubah, misalnya prosedur P yang dimaksudkan untuk menuliskan sebuah nilai N, maka N tidak pernah diubah.

Pendefinisian Prosedur	Contoh Pemanggilan Prosedur
<pre> Procedure P (N:integer); begin (* deretan instruksi, badan prosedur *) end; </pre>	<p>P(N); N adalah nama Variabel atau nama konstanta, yang type nya sama dengan type parameter P. perhatikan bahwa walaupun namanya sama yaitu N, N pada program pemanggil akan “berbeda” dengan N pada definisi prosedur.</p> <p>P(X); X adalah nama Variabel atau nama konstanta, yang type nya sama dengan type parameter P</p> <p>P(3); sebuah nilai yang sesuai type variabel</p>

Pendefinisian Prosedur	Contoh Pemanggilan Prosedur
	Parameter boleh sebuah variabel atau nilai konstanta atau konstanta sebab parameter hanya dipakai.

Parameter boleh sebuah variabel atau nilai konstanta atau konstanta sebab parameter hanya dipakai.

Jika parameter prosedur akan diubah nilainya saat keluar dari prosedur, maka sebelum nama parameter harus ditambahkan kata VAR, yang artinya *passing parameter by reference* yang memungkinkan nilai parameternya diubah.

Pendefinisian Prosedur	Pemanggilan Prosedur
<pre>Procedure UbahK (VAR K:integer); begin (* deretan instruksi, badan prosedur *) end;</pre>	<pre>K: integer; UbahK (K); Parameter harus sebuah nama variabel sebab akan diubah</pre>

Parameter harus sebuah variabel sebab akan diubah

7.8.4 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN
Pendahuluan	Pada awal pembelajaran, guru menjelaskan apa itu subprogram dalam pemrograman pascal. Terdapat 2 jenis sub program dalam pascal yaitu FUNCTION dan PROCEDURE . Untuk lebih memahami dari setiap jenis perulangan guru mengajak siswa untuk berlatih membuat subprogram.
Inti	<p>Latihan F01 – fungsi untuk menambahkan dua buah integer yang menghasilkan integer.</p> <p>Bacalah dengan cermat contoh Program P-AP-25 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre>program contohfungsi; (* contoh fungsi dan pemakaiannya *) VAR a:integer = 3; b:integer = 4; c:integer; function Tambah (a,b:integer):integer; Begin Tambah := a+b; end; begin writeln ('a= ', a); writeln ('b= ', b); c:= Tambah (a,b); writeln ('c=a+b = ', c); end.</pre>

KEGIATAN	DISKRIPSI KEGIATAN
	<p>Catatan : kelihatannya teks program mengada-ada. Namun akan banyak gunanya jika proses komputasinya tidak sesederhana itu (hanya 1 baris). Dengan hanya memanggil fungsi, keterbacaan program menjadi lebih jelas.</p> <p>Latihan P01- Prosedur tanpa parameter</p> <p>Bacalah dengan cermat contoh Program P-AP-26 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre> program Demo; (* contoh program yang mengandung prosedur *) procedure Print; begin writeln ('Dari dalam prosedur Print'); end; begin Print; end. </pre> <p>Kelihatannya tidak ada gunanya karena isi dari prosedur Print hanya 1 baris, tapi dapat dibayangkan bahwa jika sebuah prosedur isinya banyak instruksi, maka di program yang memanggilnya akan kelihatannya ringkas. Menulis prosedur dilakukan seperti menaruh teks sangat rinci di lampiran, yang hanya diacu pada teks utamanya.</p> <p>Latihan P02 – prosedur dengan parameter yang tidak diubah.</p> <p>Bacalah dengan cermat contoh Program P-AP-27 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <pre> program Demo; (*contoh program yang mengandung prosedur dengan parameter*) (* yang tidak diubah nilainya *) var bil:integer = 5; procedure Print (i:integer); begin writeln ('Nilai yang diprint', i); end; begin Print (bil); bil := 10; Print (bil); end. </pre> <p>Pada contoh di atas, Print dapat dipakai berkali-kali, dengan nilai yang dicetak sebagai parameter yang tidak diubah.</p>

KEGIATAN	DISKRIPSI KEGIATAN
	<p>Latihan P03- Prosedur dengan parameter yang merupakan hasil komputasi</p> <p>Bacalah dengan cermat contoh Program P-AP-28 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <p>Parameter yang diubah diberi awalan VAR</p> <pre> program Add2bil; VAR a:integer= 3; b:integer = 5; Result: integer; procedure Tambah (VAR Hsl:integer; a,b:integer); (* Hsl= a + b *) begin Hsl:= a + b; end; begin Tambah (Result, a,b); writeln (Hasil a+b = ', Result); end. </pre> <p>Bandingkanlah prosedur ini dengan fungsi untuk menambahkan dua buah bilangan pada latihan fungsi. Analisislah dan jelaskan perbedaannya!</p> <p>Latihan P04- Prosedur dengan parameter yang diubah nilainya</p> <p>Bacalah dengan cermat contoh Program P-AP-29 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <p>Parameter yang diubah diberi awalan VAR</p> <pre> program Demo; VAR bil:integer = 5; lo.procedure Tambahkan (VAR bil:integer; delta:integer); begin bil:= bil+delta; end; begin writeln ('nilai bil= ', bil); Tambahkan (bil, 2); writeln ('nilai bil= ', bil); Tambahkan (bil, 1); writeln ('nilai bil= ', bil); end. </pre> <p>Coba eksekusi program di atas tanpa VAR. dan bandingkan hasilnya!</p>

KEGIATAN	DISKRIPSI KEGIATAN						
	<p>Latihan P05- SWAP – menukar nilai dua buah variabel</p> <p>Bacalah dengan cermat contoh Program P-AP-30 sebagai berikut. Setelah memahami program tersebut, tuliskan programnya (koding) :</p> <p>Prosedur untuk menukar nilai dua buah variabel.</p> <pre> program Demo; (* contoh menukar nilai dari dua buah variabel *) var bill:integer = 5; bil2:integer = 10; procedure Tukar (var a:integer; var b:integer); (* menukar nilai a dan b *) var temp: integer; begin temp:= a; a:= b; b:=temp; end; begin writeln ('nilai bill= ', bill); writeln ('nilai bil2= ', bil2); Tukar (bill, bil2); writeln ('nilai bill= ', bill); writeln ('nilai bil2= ', bil2); end. </pre> <p>Pembahasan :</p> <table> <tr> <th>Pendefinisian Prosedur</th><th>Pemanggilan Prosedur</th></tr> <tr> <td> <pre> procedure Tukar (VAR a:integer; VAR b:integer); </pre> </td><td> Pendeklarasian prosedur untuk menukar nilai a dengan b. Karena akan ditukar, maka variabel a dan b harus dapat berubah. Untuk dapat berubah, harus diawali VAR </td></tr> <tr> <td> <pre> Begin temp:= a; a:= b; b:=temp; end; </pre> </td><td> Proses penukaran nilai variabel a dan b, yang membutuhkan sebuah penampung sementara bernama temp. </td></tr> </table> <p>Setelah selesai pembahasan subprogram function dan prosedur selesai, siswa diminta untuk mengerjakan soal pada Lembar Kerja siswa.</p>	Pendefinisian Prosedur	Pemanggilan Prosedur	<pre> procedure Tukar (VAR a:integer; VAR b:integer); </pre>	Pendeklarasian prosedur untuk menukar nilai a dengan b. Karena akan ditukar, maka variabel a dan b harus dapat berubah. Untuk dapat berubah, harus diawali VAR	<pre> Begin temp:= a; a:= b; b:=temp; end; </pre>	Proses penukaran nilai variabel a dan b, yang membutuhkan sebuah penampung sementara bernama temp.
Pendefinisian Prosedur	Pemanggilan Prosedur						
<pre> procedure Tukar (VAR a:integer; VAR b:integer); </pre>	Pendeklarasian prosedur untuk menukar nilai a dengan b. Karena akan ditukar, maka variabel a dan b harus dapat berubah. Untuk dapat berubah, harus diawali VAR						
<pre> Begin temp:= a; a:= b; b:=temp; end; </pre>	Proses penukaran nilai variabel a dan b, yang membutuhkan sebuah penampung sementara bernama temp.						

KEGIATAN	DISKRIPSI KEGIATAN
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisi pada form yang sudah dibuat

7.8.5 Lembar Kerja Siswa

Mengerjakan **Latihan P04 - Prosedur dengan parameter yang diubah nilainya** menggunakan kode program P-AP- 29

7.8.6 Assesment

Rubrik untuk Lembar Kerja Siswa

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program prosedur dengan/tanpa penerapan var	Program dapat berjalan dengan/tanpa menggunakan var	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak dapat berjalan dengan/tanpa menggunakan var
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengoreksi secara mandiri	Siswa sesekali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)

7.8.7 Lembar Refleksi Siswa

Aspek	Refleksi Siswa
Apakah siswa memahami dan dapat menerapkan penggunaan prosedur dan function pada program pascal?	
Apakah siswa bisa menerapkan fungsi prosedur dan function untuk menghasilkan output program ?	
Apakah siswa mampu mendeteksi kesalahan program yang tidak bisa berjalan?	
Apakah senang belajar fungsi prosedur dan function pada pemrograman dengan bahasa pascal?	

Aspek	Refleksi Siswa
Apakah terdapat kesulitan dalam belajar dalam materi tentang prosedur dan function pada bahasa pemrograman pascal?	

7.8.8 Pengayaan

1. Mistery adalah seorang programmer. Saat menulis sebuah program, semua variabelnya dideklarasikan sebagai variabel global. Tidak ada satupun sub programnya mengandung deklarasi. Menurutmu, apakah program Mistery dapat berjalan dengan baik ?
2. Lazy adalah programer yang malas menulis parameter. Programnya tidak ada satupun yang mengandung parameter. Tapi programnya dapat dieksekusi. Trick apa yang dipakainya ? apakah itu baik ?
Diskusikan, apakah kamu akan meniru Mistry atau Lazy ? Mengapa ?

7.8.9 Contoh Soal Ulangan

Perhatikan koding program berikut :

```

Program subprogram;
var
  a, b:integer;
  procedure tukar(x, y : integer);
  var tmp:integer;
  begin
    tmp:=x;
    x:=y;
    y:=tmp;
  end;
begin
  y:=1;
  x:=2;
  tukar(a, b);
  writeln('a : ', a);
  writeln('b : ', b);
end.

```

Dari program diatas jawablah pertanyaan berikut :

1. Apakah program `subprogram` tersebut bisa berjalan, jika tidak temukan kesalahan yang ada dan harus diperbaiki seperti apa
2. Tuangkan `procedure tukar` tersebut menjadi `function`, dan tunjukkan output hasilnya. Kesulitan apa yang dijumpai?

7.9 Aktivitas-9 miniproject

Kode Aktivitas : K10-AP-P8-A9-Miniproject

7.9.1 Pertanyaan Pemantik

Pernahkan kalian mendapatkan tugas fisika untuk dapat menghitung perubahan wujud suatu benda akibat suhu yang berubah? Bagaimana kalian akan membuat sebuah model perubahan wujud suatu benda akibat perubahan suhu dalam bentuk bahasa pemrograman?

7.9.2 Konsep

Konsep disini menggunakan seluruh konsep yang telah di jabarkan pada aktivitas 1 sampai dengan aktivitas 8.

7.9.3 Gambaran Umum Kegiatan

KEGIATAN	DISKRIPSI KEGIATAN
Pendahuluan	Guru membagi siswa dalam kelompok-kelompok dimana setiap kelompok berjumlah 3 anak. Kemudian siswa mencermati permasalahan, algoritma, rancangan input, rancangan output dan menjawab soal-soal latihan yang ada dibawahnya.
Inti	<p>Berupa Mini Project , sebagai penutup dari kegiatan pembelajaran pada pilar Logika Algoritma. Dalam mini projek ini sudah tersaji : Permasalahan, Input, Output dan contoh kasus untuk masukan serta beberapa latihan yang harus diselesaikan.</p> <p>Permasalahan</p> <p>Kemarin, setelah belajar Fisika tentang perubahan fasa, Rara memahami bahwa pada tekanan 1 atm, H₂O (air) itu ternyata berubah wujudnya menjadi es kalau temperaturnya di bawah 0° C, dan mulai berubah wujud menjadi water mixture dan steam saat temperatur mencapai 100° C (titik didih). Setelah itu, jika dipanaskan terus, maka temperatur akan menaik dan wujudnya adalah uap. Bu Guru hanya memberikan sebuah kurva sebagai berikut, yang memberikan gambaran lengkap wujud air jika dipanaskan mulai dari temperatur -100°C sampai 400°C pada tekanan 1 atmosfir. Bu Guru mengatakan, bahwa suhu 0° C disebut sebagai titik beku, dan pada titik beku, H₂O akan berwujud campuran es dan air. Sedangkan 100° C disebut sebagai titik penguapan, dan pada titik penguapan, H₂O akan berwujud campuran air dan uap. Bu Guru memberikan gambar sebagai berikut:</p>

KEGIATAN	DISKRIPSI KEGIATAN
	<div data-bbox="422 230 1308 958" data-label="Figure"> </div> <div data-bbox="443 981 746 1019" data-label="Section-Header"> <p><u>Keterangan gambar :</u></p> </div> <div data-bbox="440 1057 825 1285" data-label="List-Group"> <ul style="list-style-type: none"> Compressed : terkompresi Liquid : cair Saturated : titik jenuh Mixture : campuran Superheater : sangat panas Vapor : uap air </div> <div data-bbox="414 1270 1409 1693" data-label="Text"> <p>Rara penasaran, ingin mengadakan percobaan, dengan menaruh sebuah termometer pada air yang dipanaskan pada tekanan 1 atm, dan programnya akan dihubungkan dengan termometer tsb sehingga membaca nilai T, yaitu temperatur dalam derajat Celcius. Program Rara akan menuliskan wujud H₂O dari pada temperatur T, dengan tekanan yang dianggap tetap yaitu 1 atmosfer tapi mencakup fasa air di bawah 0° C . Termometer Rara hanya dapat memberikan angka bilangan bulat. Untuk sementara, karena Rara belum bisa membeli sensor yang membaca temperatur, Rara akan mencoba programnya dengan mengetikkan sebuah nilai T yang kelak akan dibaca oleh sensor. Bantulah Rara membuat program tersebut!</p> </div> <div data-bbox="414 1731 667 1767" data-label="Section-Header"> <p>Format Masukan:</p> </div> <div data-bbox="414 1771 1070 1809" data-label="Text"> <p>Sebuah bilangan bulat n, di mana $-100 < n < 400$.</p> </div> <div data-bbox="414 1845 667 1883" data-label="Section-Header"> <p>Format Keluaran:</p> </div> <div data-bbox="414 1886 1106 1924" data-label="Text"> <p>Satu baris berisi sebuah string yang berisi wujud air.</p> </div> <div data-bbox="414 1962 531 2000" data-label="Section-Header"> <p>Kasus :</p> </div>

KEGIATAN	DISKRIPSI KEGIATAN		
	Kasus	Contoh Masukan	Wujud Air
	1	-10	Es
	2	0	Campuran es dan air
	3	10	Cair
	4	100	Campuran air dan uap
	5	200	Uap
	<p>Persoalan :</p> <ol style="list-style-type: none"> 1. Kenapa Rara cukup mencoba dengan masukan tersebut ? 2. Rara bosan mengetikkan satu per satu nilai T. Kenyataannya kelak, kalau Rara berhasil membeli sensor dan program dapat membaca otomatis nilai, maka program akan membaca dari -100 s.d. 400, dan akan menuliskan setiap kenaikan 1° C. Tuliskan program yang akan mengandung pengulangan sehingga Rara tidak perlu mengetikkan satu per satu nilai T. Buatlah spesifikasi programnya, dan buatlah kasus ujinya. Catatan : yang disebut “spesifikasi perogram” adalah deskripsi Format Masukan, algoritma dan Format Keluaran yang jelas. 3. Rara tidak ingin membatasi programnya dengan 100° C s.d. 400° C. Ia akan meminta pemakai programnya yang menentukan batas nilai minimum dan maksimum temperatur tersebut. Ubahlah program pada butir (2) menjadi program yang dapat menerima batasan minimum dan maksimum. 4. Wah,tapi komputer bisa salah ya kalau bilanganterlalu kecil atau terlalu besar. Bisakah program mencegah pemakai untuk memberikan input minimum dan maksimum, tetapi tidak mengakibatkan komputer nge-<i>“hang”</i> karena nilai yang diberikan di luar batas yang dapat diterima sebuah bilangan integer. Perbaikilah program sehingga dapat menghindari programnya nge-<i>“hang”</i> 		
Penutup	Guru bersama siswa melakukan refleksi sesuai dengan materi yang telah dipelajari dan mengisikan pada form yang sudah dibuat		

7.9.4 Lembar Kerja Siswa

-

7.9.5 Lembar Refleksi Siswa

Aspek	Refleksi Guru
Dapatkah siswa mengerjakan latihan pada mini prject	
Apakah siswa dapat bekerjasama dengan kelompoknya dengan jiwa gortong royon, mandiri dan berfikir kreatif?	
Apakah siswa senang mengerjakan latihan pada project mini?	

Apakah siswa mengalami kesulitan dalam pengerjaan latihan pada project mini ?	
---	--

7.9.6 Assesment

Rublik Project mini (dibuat untuk setiap versi program Rara)

Aspek Yang dinilai	A=4	B=3	C=2	D=1
Design input-output-proses	Siswa mampu membuat 3 elemen disain input, proses dan output untuk persoalan yang diberikan	Siswa mampu membuat satu dari 2 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat satu dari 3 elemen disain (input, dan output) untuk persoalan yang diberikan	Siswa mampu membuat disain input, proses dan output untuk persoalan yang diberikan
Kesalahan sintaks.	Program dapat dikompilasi dengan baik	(tidak ada nilai B)	(tidak ada nilai C)	Program tidak lolos kompilasi
Kebenaran Program perubahan suhu	Program dapat menulis wujud akibat perubahan suhu dengan benar	(tidak ada nilai B)	(tidak ada nilai C)	Program menghasilkan output yang salah
Debugging dan testing	Saat menemukan kesalahan, siswa dapat mengkoreksi secara mandiri	Siswa sesekali minta bantuan guru/teman saat menjumpai kesalahan program	Siswa sering minta bantuan guru/teman saat menjumpai kesalahan program	Siswa selalu minta bantuan guru/teman saat menjumpai kesalahan program (tidak mandiri)
Menjawab pertanyaan	Dapat menjawab 4 pertanyaan dengan benar	Dapat menjawab 3 pertanyaan dengan benar	Dapat menjawab 2 pertanyaan dengan benar	Dapat menjawab 1 pertanyaan dengan benar

8 Referensi

Langsung diberikan di bagian akhir materi pada setiap aktivitas

9 Pengayaan

Kegiatan Pengayaan tidak disediakan dalam modul ajar ini. Banyak sekali latihan pemrograman yang disediakan di Internet, bahkan ada yang menyediakan “juri” yang akan menilai program. Dengan mempelajari deskripsi persoalannya, siswa dapat mengetik program dengan IDE yang ada, dan mengirimkan file program untuk dinilai oleh juri. Lingkungan semacam ini disebut sebagai autograder. Siswa dapat diajak untuk mempelajari dan latihan sendiri jika memang berminat. Bahan dan latihan memeriksa program ke autograder dalam bahasa Indonesia tersedia di : <https://tlx.toki.id/courses>

10 Lembar Refleksi Guru

Aspek	Refleksi Guru
Dapatkah semua siswa memahami tujuan pembelajaran dari serangkaian kegiatan pembelajaran yang telah dilakukan ?	
Efektifkah teknik pembelajaran yang digunakan ?	
Apakah seluruh siswa dapat mengikuti pembelajaran dengan baik ?	
Apakah guru juga suka dan senang memrogram ?	
Apakah guru mengalami kesulitan dalam memprogram ?	
Apakah guru akan membuat program untuk membantu kegiatan sehari-harinya ?	

11 Pesan Pesan Pedagogi Perancang Modul Untuk Guru

Pemrograman merupakan salah satu pilar informatika yang sangat penting. Pemrograman dapat menjadi menarik jika siswa tidak diajak hanya memahami sintaks, tetapi memahami struktur program serta memahami maknanya. Pemrograman didasari oleh algoritma yang merupakan salah satu dari empat fondasi Computational Thinking.

Guru dapat mengembangkan metode pembelajaran sesuai dengan karakteristik siswa yang diajar dan menambahkan pengembangan pada kasus-kasus yang digunakan sebagai contoh dan latihan.