2. Variable, Expression dan Statements

2.1 Tujuan Praktikum

Setelah mempelajari Bab ini, mahasiswa diharapkan dapat:

1. Dapat menjelaskan dan menggunakan variabel, ekspresi, dan statement pada Python.

2.2 Alat dan Bahan

Praktikum ini membutuhkan perangkat komputer yang memiliki spesifikasi minimum sebagai berikut:

1. Terkoneksi ke Internet dan dapat mengunduh package-package Python.
2. Mampu menjalankan sistem operasi Windows 10 atau Ubuntu Linux.

Perangkat lunak yang diperlukan untuk mendukung praktikum ini adalah sebagai berikut:

1. Python 3.7 atau 3.8 yang terinstall menggunakan Anaconda atau Installer Python lainnya.
2. Editor Visual Studio Code, Repl Python 3 (online), pyCharm, atau Spyder

2.3 Materi

2.3.1 Values dan type

Value merupakan komponen utama dari program, seperti huruf atau angka. *Value* yang sering kita kenal misalnya 1,2,'a','z', dan "Hello Word". *Value* dibagi menjadi beberapa tipe yang berbeda, misalnya 2 untuk sebuah nilai integer dan "Hello Word" untuk sebuah nilai *string*. Interpreter dapat melakukan identifikasi terhadap *string* karena pada penulisannya ditutup menggunakan tanda petik (*quation mark*).

Untuk mecoba memahami values dan type, silakan coba beberapa baris kode berikut ini menggunakan python interactive mode seperti yang ditunjukkan pada potongan kode dibawah ini.

```
>>> print(4)
4
>>> print(10.876)
10.876
```

```
>>> print('Z')
Z
>>> print('True')
True
>>> print('False')
False
```

Perintah print juga bekerja untuk value selain *string*, seperti *integer* (bilangan bulat), *float* (bilangan pecahan), *character* (huruf), atau *bool* (benar/salah). Untuk mencobanya kita dapat menggunakan perintah python untuk menjalankan interpreter.

Setiap value yang pasti memiliki type untuk mengetahui tipe data tersebut. Python menyediakan fungsu built-in untuk melakukan pegecekan tipe data pada value dengan menggunakan fungsi *type()*

```
>>> x=5
>>> print(x, "tipenya adalah ", type(x))
5 tipenya adalah <class int'>
>>> x = 2.0
>>> print(x, "tipenya adalah ", type(x))
2.0 tipenya adalah <class float'>
>>> x = 1+2j
>>> print(x, "tipenya adalah ",type(x))
(1+2j) tipenya adalah <class complex">
```

Ketika menggunakan bilangan bulat besar, beberapa model penulisan menggunakan tanda koma (,) diantara kelompok tiga digit. Misalnya pada penulisan 1.000.000. Dalam python, akan dianggap sebagai bilangan bulat.

```
>>> print(1,000,000)
1,0,0
```

Hal ini terjadi karena Python menganggap bahwa 1,000,000,000 merupakan kiriman parameter sebanyak 3 parameter pada fungi print, yaitu 1, 0, dan 0.

2.3.2 Variabel

Salah satu fitur powerfull dalam bahasa pemrograman adalah kemampuannya untuk melakukan manipulasi *variable*. *Variable* merupakan lokasi memori yang dicadangkan untuk menyimpan nilai-nilai. Ini berarti bahwa ketika Anda membuat sebuah *variable* Anda memesan beberapa ruang di memori. *Variable* menyimpan data yang dilakukan selama program dieksekusi, yang natinya isi dari variabel tersebut dapat diubah oleh operasi - operasi tertentu pada program yang menggunakan *variable*.

```
>>> pesan = 'selamat pagi, mari belajar python'
>>> n = 17
>>> pi = 3.1415926535897931
```

Variable dapat menyimpan berbagai macam tipe data. Di dalam pemrograman Python, *variable* mempunyai sifat yang dinamis, artinya *variable*Python tidak perlu didekralasikan tipe data tertentu dan *variable* Python dapat diubah saat program dijalankan.

Potongan kode diatas merupakan contoh penggunaan dari *varibale*. Contoh pertana adalah variable pesan yang berisi string, contoh kedua adalah variale n yang berisi nilai integar 17 dan contoh ketiga merupakan nilai dari pi (π). Untuk menampilkan nilai dari varibale, dapat digunakan perintah print.

```
>>> print(n)
17
>>> print(ipk)
3.29
```

2.3.3 Nama Variabel dan Keywords

Pemberian nama pada variabel mengacu pada panduan berikut ini.

1. Nama variable boleh diawali menggunakan huruf atau garis bawah (_), contoh: nama, _nama, namaKu, nama_variable.
2. Karakter selanjutnya dapat berupa huruf, garis bawah (_) atau angka, contoh: _nama, n2, nilai1.
3. Karakter pada nama variable bersifat sensitif (*case-sensitif*). Artinya huruf besar dan kecil dibedakan. Misalnya, variabel_Ku dan variabel_ku, keduanya adalah variabel yang berbeda.
4. Nama variabel tidak boleh menggunakan kata kunci yang sudah ada dalam python seperti if, while, for, dsb.

Python sendiri memiliki 35 *keyword* yang tidak boleh digunakan untuk memberi nama variabel.

and	**del**	**from**	**None**	**True**
as	**elif**	**global**	**nonlocaly**	**try**
assert	**else**	**if**	**not**	**while**
break	**except**	**import**	**or**	**width**
class	**False**	**in**	**pass**	**yield**
continue	**finally**	**is**	**raise**	**async**
def	**for**	**lamda**	**return**	**wait**

Berikut ini contoh penggunaan variable dalam bahasa pemrograman Python.

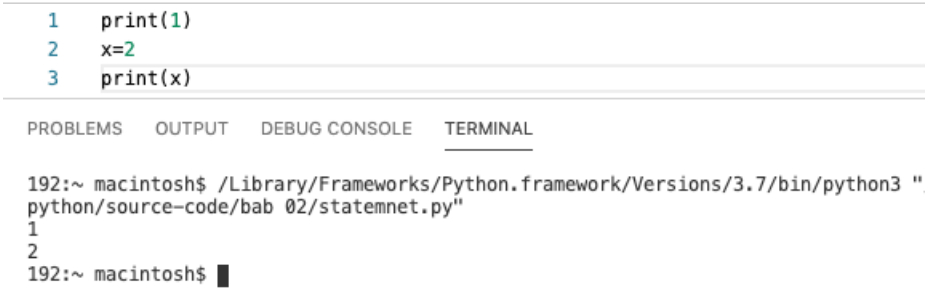
```
#proses memasukan data ke dalam variabel
nama = "Agung Sejagat"

#proses mencetak variabel
print(nama)

#nilai dan tipe data dalam variabel  dapat diubah
umur = 20
print(umur)
type(umur)
umur = "dua puluh satu" #nilai setelah diubah
print(umur)             #mencetak nilai umur
type(umur)              #mengecek tipe data umur
namaDepan = "Joko"
namaBelakang = "Widodo"
nama = namaDepan + " " + namaBelakang
umur = 22
hobi = "Berenang"
print("Biodata\n", nama, "\n", umur, "\n", hobi)

#contoh variabel lainya
inivariabel = "Halo"
```

```
ini_juga_variabel = "Hai"
_inivariabeljuga = "Hi"
inivariabel222 = "Bye"
panjang = 10
lebar = 5
luas = panjang * lebar
print(luas)
```

2.3.4 Statements

Statements merupakan bagian dari code interpreter Python yang dapat dieksekusi. Misalnya pada statement print, dapat berupa *expression statements* dan *assignment*.

Ketika menggunakan pyhton dalam mode interaktif, interpreter secara langsung akan melakukan eksekusi dan menampilkan hasilnya. Hal ini tentu saja berbeda ketika menggunakan *script mode*. *Script* biasanya berisi *statements* yang saling berhubungan secara sekuensial.



Gambar 2.1: Contoh statement dan outputnya

2.3.5 Operator dan Operand

Operator adalah simbol tertentu yang digunakan untuk melakukan operasi aritmatika maupun logika. Nilai yang padanya dilakukan operasi disebut *operand*. Misalnya adalah 2 + 3. Di sini tanda + adalah operator penjumlahan. 2 dan 3 adalah operand.

Pada bagian ini secara khusus akan membahas operator aritmatika pada Python. Operator aritmatika adalah operator yang digunakan untuk melakukan operasi matematika, seperti penjumlahan, pengurangan, perkalian, pembagian, dan sebagainya. Tabel berikut menunjukkan jenis operator aritmatika.

Tabel 2.1: Operator pada Python

Operator	Nama dan Fungsi	Contoh
+	Penjumlahan, menjumlahkan 2 buah operand	x + y
–	Pengurangan, mengurangkan 2 buah operand	x - y
*	Perkalian, mengalikan 2 buah operand	x * y
/	Pembagian, membagi 2 buah operand	x / y
**	Pemangkatan, memangkatkan bilangan	x **y

Beberapa contoh penggunaan operator artimatika pada Python dapat dilihat pada potongan kode berikut ini

```
>>>32 + 30
62
```

```
>>> hour = 5
>>> print (hour-1)
4
>>> minute=60
>>> print (hour*6+minute)
90
>>> print (minute/60)
1.0
>>> 5**2
25
>>> (5+9)*(15-7)
112
```

2.3.6 Expressions

Expression merupakan representasi dari nilai dan dapat terdiri dari gabungan antara *values*, *variable* dan *operator*. *Values* dengan sendirinya dapat dianggap sebagai *expression* dan juga variabel. Secara umum, semuanya dapat disebut dengan *expression*.

```
17
x
x+17
```

Ketika menggunakan expression dalam model intercative, interpreter akan melakukan evaluasi dan menampilkan hasilnya.

```
>>> 1 + 1
2
>>> 3 + 2
5
```

2.3.7 Urutan Operasi

Urutan operasi berlaku bila ada lebih dari satu operator dalam expression. Urutan operasi bergantung pada aturan prioritas. Untuk operasi matematika, Python mengikuti konversi matematika. Urutan operasi sering disingkat dengan **PEMDAS - Parantheses, Exponentiation, Multiplication and Division, Operator**.

- *Paranthese* (Tanda kurung) - merupakan prioritas tertinggi dan digunakan untuk memaksa expression dalam urutan yang sesuai. Contohnya 2*(3-1) hasilnya 4, dan (1+1) ** (5-2) hasilnya 8. Penggunaan tanda kurung dapat digunakan untuk membuat expression menjadi lebih mudah untuk dibaca, misalny (minute * 100) / 60.
- *Exponentiation* (Eksponensial/Pemangkatan) - merupakan prioritas tertinggi berikutnya, contoh 2**1+1 hasilnya 3, bukan 4, dan 3*1**3 hasilnya 3 bukan 27
- *Multiplication and Divison* (Perkalian dan Pembagian) - memiliki prioritas yang sama tetapi lebih tinggi dari penjumlahan dan pengurangan. Penjumlahan dan pengurangan juga memiliki prioritas yang sama pula. Contoh 2*3-1 hasilnya 5 bukan 4, dan 6+4/2 hasilnya 8, bukan 5.
- *Operators* - operator memiliki prioritas yang sama, dibaca dari kiri ke kanan. Contoh 5-3-1 hasilnya 1 bukan 3 karena operasi pengurangan 5-3 terlebih dahulu baru kemudian hasilnya dikurangi dengan 1.

Jika terjadi keraguan, silakan letakkan tanda kurung di dalam ekspresi untuk memastikan bahwa komposisinya sesuai dengan yang diinginkan.

2.3.8 Operator Modulus dan String

Modulus

Operator Modulus merupakan sisa hasil bagi dari bilangan pertama dengan bilangan kedua. Operator ini hanya berlaku pada tipe data integer. Dalam python, operator modulus dilambangakan dengan tanda persen (%).

```
>>> quotient = 7 // 3
>>> print(quotient)
2
>>> oprmomulus = 7 % 3
>>> print(oprmomulus)
1
```

7 dibagi dengan 3 menghasilkan 2 dnegan sisa hasil bagi 1.

Contoh penggunaan operator modulus.

- Memeriksa satu angka dapat dibagi dengan yang lain, misal jika x % y adalah 0, maka x dapat dibagi oleh y.
- Dapat mengekstrak digit paling kanan atau digit dari suatu angka. Misalnya, x% 10 menghasilkan digit x paling kanan (dalam basis 10). Demikian pula, x% 100 menghasilkan dua digit terakhir.

String

Operator + ketika bekerja dengan string tidak berarti penjumlahan secara matematika, melainkan penggabungan antar string. Contoh:

```
>>> first = 10
>>> second = 15
>>> print(first+second)
25
>>> first = '100'
>>> second = '150'
>>> print(first + second)
100150
```

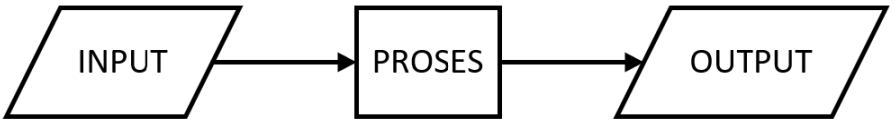
Operator * juga bekerja dengan string dengan melakukan perkalian antara content string dan integer.

```
>>> first = 'Test '
>>> second = 3
>>> print(first * second)
Test Test Test
```

2.3.9 Menangani Input dari Pengguna

Sebuah program biasanya memiliki alur kerja Input - Proses - Output yang alurnya ditunjukkan pada Gambar 2.2.

Input adalah data/masukan yang dibutuhkan supaya program bisa berjalan. Proses adalah langkah-langkah yang dilakukan oleh program untuk memecahkan masalah. Sedangkan Output adalah hasil yang didapatkan setelah menjalankan langkah-langkah tersebut. Sebagai contoh misalnya mengambil uang lewat ATM. Pengambilan uang melalui ATM dapat dibagi menjadi 3 bagian (Input-Proses-Output) tersebut, yaitu:



Gambar 2.2: Input - Proses - Output.

1. Masukkan kartu ATM. Masukkan PIN anda. Kartu ATM dan PIN adalah Input yang diperlukan supaya anda bisa mengambil uang.
2. Anda memilih menu Pengambilan Uang. Kemudian anda memasukkan nominal yang diinginkan. Bagian ini juga merupakan Input.
3. Mesin ATM akan memproses transaksi anda dengan menghubungi server Bank yang bersangkutan. Dilakukan berbagai macam pengecekan (misal: apakah saldonya cukup? apakah kartunya masih berlaku? apakah ada blokir?). Bagian ini disebut Proses.
4. Mesin ATM mengeluarkan uang, bukti pengambilan uang dan kartu anda. Uang yang keluar merupakan Output dari kegiatan ini. Selain itu, saldo anda juga berkurang. Pengurangan saldo juga merupakan hasil dari kegiatan ini.

Python juga dapat menangani input dari pengguna. Input dalam hal ini dapat berupa input text yang dimasukkan oleh pengguna. Untuk itu python menyediakan built-in fucntion yang disebut input untuk mendapatkan input dari keyboard. Ketika fungsi ini dipanggil, program akan berhenti dan menunggu pengguna untuk mengetik sesuatu. Ketika pengguna menekan tombol Enter, program akan dilanjutkan dan input akan mengambalikan apa yang diketik oleh pengguna sebagai string.

```
>>> inp = input()
Pada hari minggu kuturut ayah ke kota
>>> print(inp)
Pada hari minggu kuturut ayah ke kota
```

Sebelum mendapatkan input dari pengguna, lebih baik untuk mencetak prompt yang memberitahu pengguna apa yang harus diinput. String tersebut dapat diteruskan ke input untuk ditampilkan kepada pengguna sebelum berhenti untuk input.

```
>>> name = input('Siapa nama mu ?\n')
Siapa nama mu ?
Sancaka
>>> print(name)
Sancaka
```

Tanda \n pada akhir prompt mewakili baris baru atau ganti baris sehingga input pengguna muncul dibawah prompt.

Ketika mengharapkan pengguna untuk mengetik bilangan bulat, dapat dilakukan dengan mengonversi nilai kembali ke int menggunakan fungsi int():

```
>>> prompt = 'Berapa suhu ruangan sekarang?\n'
>>> suhu = input(prompt)
Berapa suhu ruangan sekarang?
24
>>> int(suhu)
```

```
24
>>> int(suhu) + 5
29
```

Akan terjadi error jika pengguna memasukkan data selain angka.

```
>>> prompt = 'Berapa suhu ruangan sekarang?\n'
>>> suhu = input(prompt)
Berapa suhu ruangan sekarang?
Ruangan depan atau belakang ?
>>> int(suhu)
ValueError: invalid literal for int() with base 10:
```

2.4 Komentar

Tanda pagar (#) digunakan untuk menandai komentar di python. Komentar tidak akan diproses oleh interpreter Python. Komentar hanya berguna untuk programmer untuk memudahkan memahami maksud dari kode.

```
# Komentar pertama
print("hai dunia!") # Komentar kedua
```

Outputnya

```
hai dunia
```

Python tidak memiliki fitur komentar multibaris. Kita harus mengomentari satu persatu baris seperti berikut:

```
# Ini komentar
# Ini juga adalah komentar
# Ini juga masih komentar
```

2.5 Kegiatan Praktikum

Kegiatan praktikum yang akan dilakukan adalah sebagai berikut:

1. Membuat variabel.
2. Memberikan nilai dalam variabel.
3. Mencetak nilai dalam variabel.
4. Separator, tipe data dan fungsi type

2.5.1 Membuat Variabel

Misalkan sebuah data pribadi berisi nama, alamat, umur, tempat lahir, tanggal lahir, indeks prestasi kumulatif akan memberikan 6 (enam) buah variabel dengan tipe datanya.

```
1 Nama = input("Nama Anda : ")
2 Alamat = input("Alamat Tinggal Anda : " )
3 Umur = input("Umur Anda : ")
4 TL = input("Tempat Lahir Anda : ")
5 Tgl = input("Tanggal Lahir Anda : ")
```

```
IPK = input("IPK Anda : ")
print("====================================")
print("DATA AKADEMIK")
print("Nama Anda : ", Nama)
print("Alamat Tinggal Anda : ",Alamat)
print("Umur Anda : ", Umur)
print("Tempat Lahir Anda : ", TL)
print("Tanggal Lahir Anda : ", Tgl)
print("IPK Anda :", IPK)
```

Output dari program diatas dapat dilihat pada gambar 2.3

```
192:~ macintosh$ /Library/Frameworks/Python.framework/Versions/3.7/bin/python3 "/Users/macintosh/Documents/GitHub/modul-a
ode/bab 02/lat_input_output.py"
Nama Anda : Wahyu Sardono
Alamat Tinggal Anda : Klaten
Umur Anda : 17
Tempat Lahir Anda : Klaten
Tanggal Lahir Anda : 19 Agustus 1986
IPK Anda : 9
====================================
DATA AKADEMIK
Nama Anda :  Wahyu Sardono
Alamat Tinggal Anda :  Klaten
Umur Anda :  17
Tempat Lahir Anda :  Klaten
Tanggal Lahir Anda :  19 Agustus 1986
IPK Anda : 9
192:~ macintosh$
```

Gambar 2.3: Contoh input/output tipe data string.

Bagaimana dengan tipe data yang lain? Kita akan mencoba untuk membuat program dengan menggunakan tipe data bilangan baik integer maupun float.

```
# CONTOH PROGRAM
# MENGHITUNG BILANGAN
#====================================

x1 = eval(input("X1 = "))
x2 = eval(input("X2 = "))
x3 = eval(input("X3 = "))
x4 = eval(input("X4 = "))

jumlah = x1+x2+x3+x4
kali = x1*x2*x3*x4
print('Hasil Penjumlah semua bilangan = ', jumlah)
print('Hasil Perkalian semua bilangan = ', kali)
jumlah = jumlah + 0.5
print('Jika ditambah 0.5 hasilnya = ',jumlah)
kali = kali * 0.5
print('Jika dikali 0.5 hasilnya = ',kali)
```

Output dari program diatas dapat dilihat pada gambar 2.4

```
didanendya:~ macintosh$ /Library/Frameworks/Python.framework/Versions/3.7/bin/python3 "/Us
ource-code/bab 02/hitung_bil.py"
X1 = 2
X2 = 5
X3 = 7
X4 = 9
Hasil Penjumlah semua bilangan =  23
Hasil Perkalian semua bilangan =  630
Jika ditambah 0.5 hasilnya =  23.5
Jika dikali 0.5 hasilnya =  315.0
```

Gambar 2.4: Contoh input/output tipe data bilangan.

Dari gambar 2.4 dapat diketahui bahwa input/output pada tipe data bilangan dengan hasil yang berbeda tipe bilangannya yaitu tipe integer (bilangan bulat) atau float (bilangan berkoma).

eval() digunakan untuk melakukan konversi expression dari **string** menjadi **integer**.

2.5.2 Memberikan nilai dalam variabel

Lakukan inisiasi variabel atau konstanta dari permasalahan berikut! Menjumlahkan total harga pada saat konsumen membeli beberapa barang.

Langkah 1 : Inisasi Persoalan

Variabel/konstanta input :
kode_barang, nama_barang, harga_satuan_barang,
jumlah_per_barang_beli, total_harga_per_transaksi = 0 Proses :
harga_beli_per_barang = harga_satuan_barang * jumlah_per_barang_beli
total_harga_per_transaksi=harga_beli_per_barang + total_harga_per_transaksi
Output :
total_harga_per_transaksi

Langkah 2 : Menetapkan Tipe Data

kd_brg, nama_brg bertipe data *string*
jum_brg bertipe data *integer*
harga_satuan, harga_beli, total_hrg_brg bertipe data *float*.

Langkah 3 : Kode Program

```
1 total_hrg_brg= 0.0
2 kd_brg=input("Kode barang = ")
3 nama_brg=input("Nama barang = ")
4 harga_satuan=eval(input("Harga satuan barang =Rp. "))
5 jum_brg=eval(input("Jumlah barang yang dibeli = "))
6 harga_beli = harga_satuan * jum_brg
7 total_hrg_brg= harga_beli + 123total_hrg_brg
8 print("Total harga yang dibayar Rp",total_hrg_brg)
```

Output dari program diatas dapat dilihat pada gambar 2.5

```
192:bab 02 macintosh$ python3 lat_variabel.py
Kode barang = 445567
Nama barang = Gunting Kuku
Harga satuan barang =Rp. 1750
Jumlah barang yang dibeli = 20
Total harga yang dinayar Rp 35000.0
192:bab 02 macintosh$
```

Gambar 2.5: Contoh input/output menggunakan variabel.

2.5.3 Mencetak nilai dalam variabel

Mencetak nilai dalam sebuah variabel menggunakan perintah print.

```
x=30
print(x)
type(x)

y = 4*int(x)
print(y)

print("Tipe data X dikonversi ke int agar dapat dihitung Y=4*x")

z = y+float(x)
print("z",z)

print("Tipe data X dikonversi menjadi float untuk dijumlahkan
dengan tipe data Y, hasilnya yang ditampung
bertipe float juga")

P=True
L=False

P!=L
print("Tanda != artinya tidak sama dengan")

P==L
print("Tanda == artinya sama dengan")
```

2.5.4 Separator, tipe data dan fungsi type

Konversi type data pada pemrograman python gunakan fungsi berikut :

1. **str()** = Untuk konversi type data ke String
2. **int()** = Untuk konversi type data ke Integer
3. **float()** = Untuk konversi type data ke Float

Ada dua macam variasi print :

1. Jika ada simbol, gunakan kutip dua atau gunakan backslash (\) sebelum menuliskan simbol.
2. Dipisahkan dengan tanda koma.

3. Diganti dengan :
    - %d : mewakili integer
    - %f : mewakili float
        - Untuk membuat n angka di belakang koma, gunakan %.nf
        - Misal untuk dua angka di belakang koma, berarti gunakan %.2f
    - %s : mewakili string

```
1 Kode='AB123'
2 NamaBarang='Kaca Mata'
3 HargaSatuan=125000
4 Stok=10
5 print('Kode barang %s \n Nama Barang= %s \n Harga Satuan=Rp %d \n Stok Barang=%d' %
6 (Kode,NamaBarang,HargaSatuan,Stok))
7
8 HargaBarang=float(HargaSatuan)
9 print('Harga Satuan Barang= Rp. %.3f' %(HargaBarang))
```

Hasil dari potongan kode program diatas dapat dilihat pada gambar 2.6

```
didanendya:~ macintosh$ /Library/Frameworks/Python.framework/Versions/3.7/bin/python3 "/Users/macintosh/Documents/GitHub
ource-code/bab 02/coba_tipe_data.py"
 Kode barang AB123
 Nama Barang= Kaca Mata
 Harga Satuan=Rp 125000
 Stok Barang=10
 Harga Satuan Barang= Rp. 125000.000
didanendya:~ macintosh$
```

Gambar 2.6: Tampilan Contoh print Tipe Data String, Integer dan Float.

2.6 Latihan Mandiri

Latihan 2.1 Buatlah program yang dapat menghitung berat badan yang diperlukan, jika diketahui tinggi badan dan nilai Body Mass Index (BMI) yang diharapkan! Body Mass Index dihitung dengan cara: $BMI = \frac{berat}{tinggi^2}$. Perhatikan, berat badan dalam satuan kilogram (kg) dan tinggi badan dalam satuan meter (m). ■

Latihan 2.2 Buatlah program yang dapat menghitung hasil dari fungsi $f(x) = 2x^3 + 2x + \frac{15}{x}$, di mana x merupakan bilangan bulat yang dimasukkan oleh pengguna. ■

Latihan 2.3 Budi tertarik untuk melamar pekerjaan pada liburan semester yang akan berlangsung selama 5 minggu. Gaji yang diberikan adalah gaji per jam. Total pajak yang harus budi bayarkan dari penghasilannya selama bekerja adalah 14%. Setelah membayar pajak, budi menghabiskan 10% dari pendapatan bersihnya untuk membeli baju dan aksesoris yang akan digunakan pada semester baru, dan 1% untuk membeli alat tulis. Setelah membeli baju, aksesoris dan alat tulis, Budi menggunakan 25% dari sisa uangnya untuk disedekahkan. Setiap Rp.1000 yang Budi sedekahkan 30% nya akan diserahkan kepada anak yatim, dan sisanya akan diserahkan ke kaum dhuafa.

Buatlah sebuah program, dengan input:

1. Gaji per jam yang anda inginkan
2. Jumlah jam kerja yang akan dilakukan dalam 1 minggu

Output dari program adalah sebagai berikut :

1. Pendapatan Budi selama libur musim panas sebelum melakukan pembayaran pajak.
2. Pendapatan Budi selama libur musim panas setelah melakukan pembayaran pajak.
3. Jumlah uang yang akan Budi habiskan untuk membeli pakaian dan aksesoris.
4. Jumlah uang yang akan Budi habiskan untuk membeli alat tulis.
5. Jumlah uang yang akan Budi sedekahkan.
6. Jumlah uang yang akan diterima anak yatim.
7. Jumlah uang yang akan diterima kaum dhuafa.

■