ALGORITMA DAN STRUKTUR DATA

INF1083

LAPORAN PRAKTIKUM 1 : Data Numerik

Oleh :

Akhmad Qasim

2211102441237

Teknik Informatika
Fakultas Sains & Teknologi
Universitas Muhammadiyah Kalimantan Timur

Samarinda, 2023

Laporan Praktikum 1:
Data Numerik

Pokok Bahasan:

- Review *Python* 3
- Tipe Data Numerik

Tujuan Pembelajaran:

- ✓ Meninjau kembali Bahasa Pemrograman *Python* 3.
- ✓ Mengenal tipe data numerik.
- ✓ Menggunakan *syntax* numerik *Python* 3.

Tugas Pendahuluan:

1. int(x, base)
   Sintaks int(x, base) adalah fungsi Python bawaan yang mengubah angka atau string dari angka menjadi bilangan bulat. Argumen pertama x adalah angka atau string yang ingin diubah menjadi bilangan bulat. Basis argumen kedua (opsional) adalah basis representasi angka dalam x dan nilai standarnya adalah 10. Jika basis ditentukan, x harus berupa representasi string dari angka dalam basis yang diberikan. Misalnya, int("1010", 2) akan mengubah string biner "1010" menjadi bilangan bulat desimal 10, dan int("10", 16) akan mengubah string heksadesimal "10" menjadi bilangan bulat desimal 16.

2. float(x)
   Sintaks float(x) adalah fungsi Python bawaan yang mengubah representasi angka atau string dari angka menjadi angka floating-point. Argumen x bisa berupa bilangan bulat, representasi string dari angka titik-mengambang, atau tipe lain yang bisa dikonversi menjadi pelampung. Jika argumennya bukan angka atau string yang dapat diubah menjadi float, pengecualian ValueError akan dimunculkan. Misalnya, float(10) akan mengonversi bilangan bulat 10 menjadi bilangan titik-mengambang 10.0, dan float("3.14") akan mengubah string "3.14" menjadi bilangan titik-mengambang 3.14.

3. complex(real[,imag])
   Kompleks sintaksis(nyata[, imag]) adalah fungsi Python bawaan yang membuat bilangan kompleks dari bagian nyata dan bagian imajiner opsional. Argumen real adalah bilangan (integer atau floating-point) yang mewakili bagian real dari bilangan kompleks. Argumen opsional imag (default 0) adalah angka yang mewakili bagian

imajiner dari bilangan kompleks. Misalnya, kompleks(1, 2) membuat bilangan kompleks dengan bagian riil 1 dan bagian imajiner 2, yang dapat direpresentasikan sebagai 1 + 2j, dan kompleks(3) membuat bilangan kompleks dengan bagian riil 3 dan bagian imajiner 0, yang dapat direpresentasikan sebagai 3 + 0j.

Percobaan & Latihan:

1.

1.1. Ha = x + y

```
4  Ha = x + y
5  print("Ha =", Ha)
```

x dan y masing-masing diberi nilai 12 dan 37. Kemudian, Ha = x + y menjumlahkan nilai x dan y dan memberikan hasilnya ke variabel Ha. Jadi, Ha = 12 + 37 = 49. Artinya nilai Ha memiliki nilai 49.

1.2. Hb = Ha + 10.5

```
7  Hb = Ha + 10.5
8  print("Hb =", Hb)
```

```
Hb = 59.5
```

Variable Hb = Ha + 10.5 mengambil nilai dari variabel Ha (yang memiliki nilai 49) dan menambahkan 10.5 ke dalamnya, lalu menugaskan hasilnya ke variabel Hb. Di sini, Ha memiliki nilai 49, yang merupakan bilangan bulat. Operator penjumlahan + menjumlahkan nilai Ha dan 10,5, yang merupakan bilangan floating. Hasil dari operasi ini akan berupa bilangan floating-point dan disimpan dalam variabel Hb. Jadi, Hb = 49 + 10,5 = 59,5. Artinya variable Hb akan bernilai 59,5.

1.3. Hc = int(Hb)

```
10  Hc = int(Hb)
11  print("Hc =", Hc)
```

```
Hc = 59
```

Variable Hb = 59.5 merupakan bilangan floating-point. Variable Hc = int(Hb) mengubah bilangan floating-point menjadi bilangan bulat menggunakan fungsi int(), dan memberikan hasilnya ke variabel Hc. Di Python, fungsi int() dibulatkan ke bawah ke bilangan bulat terdekat jika argumennya adalah angka floating. Jadi, Hc = int(59.5) = 59. Artinya variable Hc akan bernilai 59.

1.4. Hd = Ha / Hb

```
13  Hd = Ha / Hb
14  print("Hd =", Hd)
```

```
Hd = 0.8235294117647058
```

Variable Ha memiliki nilai 49, yang merupakan bilangan bulat. Variable Hb memiliki nilai 59,5, yang merupakan bilangan floating-point. Kode Hd = Ha / Hb membagi nilai Ha dengan nilai Hb dan menugaskan hasilnya ke variabel Hd. Karena Ha dan Hb adalah nilai numerik, maka nilainya akan diubah menjadi floating dan mengembalikan hasil floating. Jadi, Hd = 49,0 / 59,5 = 0.8235294117647058. Ini berarti variable Hd akan menjadi 0.8235294117647058.

1.5. isinstance(Hd, int)

```
16  print(isinstance(Hd, int))
```

```
False
```

Kode isinstance(Hd, int) di Python memeriksa apakah nilai yang disimpan dalam variabel Hd adalah turunan dari tipe int (integer). Fungsi isinstance() mengambil dua argumen: argumen pertama adalah nilai yang ingin kita periksa tipenya, dan argumen kedua adalah tipe yang ingin kita periksa. Dalam hal ini, Hd adalah nilai yang ingin kita periksa, dan int adalah tipe yang ingin kita periksa. Jika Hd bilangan bulat, maka isinstance(Hd, int) akan mengembalikan True. Jika tidak, jika Hd bukan bilangan bulat, maka isinstance(Hd, int) akan mengembalikan False. Karena Hd memiliki nilai 0.8235294117647058, maka akan mengeluarkan output false.

1.6. isinstance(Hd, float)

```
18  print(isinstance(Hd, float))
```

```
True
```

Kode isinstance(Hd, float) di Python memeriksa apakah nilai yang disimpan dalam variabel Hd adalah turunan dari tipe float (floating-point number). Fungsi isinstance() mengambil dua argumen: argumen pertama adalah nilai yang ingin kita periksa tipenya, dan argumen kedua adalah tipe yang ingin kita periksa. Dalam hal ini, Hd adalah nilai yang ingin kita periksa, dan float adalah tipe yang ingin kita periksa. Jika Hd adalah bilangan floating-point, maka isinstance(Hd, float) akan mengembalikan True. Jika tidak, jika Hd bukan angka floating, maka isinstance(Hd, float) akan mengembalikan False. Karena Hd memiliki nilai 0.8235294117647058, maka akan mengeluarkan output true.

1.7.
```
if isinstance(Hd, int):
    print("Hd adalah bilangan integer")
else:
    print("Hd adalah bilangan float")
```
```
Hd adalah bilangan float
```
Kode dalam Python ini memeriksa jenis nilai yang disimpan dalam variabel Hd. Variable Hd memiliki nilai 0,8235294117647058, yang merupakan bilangan floating-point. Kode menggunakan fungsi isinstance() untuk memeriksa apakah Hd merupakan turunan dari tipe int (integer). Jika Hd adalah bilangan bulat, maka isinstance(Hd, int) akan mengembalikan True dan kode di dalam blok pertama akan dieksekusi. Jika tidak, maka isinstance(Hd, int) akan mengembalikan False dan kode di dalam blok kedua (blok else) akan dieksekusi. Dalam hal ini, kode akan menampilkan "Hd adalah bilangan float".

2.

2.1. (999).bit_length()
```
print((999).bit_length())
```
```
10
```
Ekspresi (999).bit_length() dalam Python mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner. Metode bit_length() adalah metode bawaan di Python yang mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner, tidak termasuk angka nol di depan. Metode ini dipanggil pada objek bilangan bulat dan mengembalikan nilai bilangan bulat. Dalam hal ini, 999 adalah bilangan bulat, dan ekspresi (999).bit_length() memanggil metode bit_length() pada bilangan bulat ini. 999 direpresentasikan dalam biner sebagai 0b1111100111, yang panjangnya 10 bit. Jadi, ekspresi (999).bit_length() akan bernilai 10.

2.2. (998).bit_length()
```
print((998).bit_length())
```
```
10
```
Ekspresi (998).bit_length() dalam Python mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner. Metode bit_length() adalah metode bawaan di Python yang mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner, tidak termasuk angka nol di depan. Metode ini dipanggil pada objek bilangan bulat dan mengembalikan nilai bilangan bulat. Dalam hal ini, 998 adalah bilangan bulat, dan ekspresi (998).bit_length() memanggil metode bit_length() pada bilangan bulat ini. 998 direpresentasikan dalam biner sebagai 0b1111100110, yang

panjangnya 10 bit. Jadi, ekspresi (998).bit_length() akan bernilai 10.

2.3. (99).bit_length()

```
5  print((99).bit_length())
```

```
7
```

Ekspresi (99).bit_length() dalam Python mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner. Metode bit_length() adalah metode bawaan di Python yang mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner, tidak termasuk angka nol di depan. Metode ini dipanggil pada objek bilangan bulat dan mengembalikan nilai bilangan bulat. Dalam hal ini, 99 adalah bilangan bulat, dan ekspresi (99).bit_length() memanggil metode bit_length() pada bilangan bulat ini. 99 direpresentasikan dalam biner sebagai 0b1100011, yang panjangnya 7 bit. Jadi, ekspresi (99).bit_length() akan bernilai 7.

2.4. (11).bit_length()

```
7  print((11).bit_length())
```

```
4
```

Ekspresi (11).bit_length() dalam Python mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner. Metode bit_length() adalah metode bawaan di Python yang mengembalikan jumlah bit yang diperlukan untuk mewakili bilangan bulat dalam format biner, tidak termasuk angka nol di depan. Metode ini dipanggil pada objek bilangan bulat dan mengembalikan nilai bilangan bulat. Dalam hal ini, 11 adalah bilangan bulat, dan ekspresi (11).bit_length() memanggil metode bit_length() pada bilangan bulat ini. 11 direpresentasikan dalam biner sebagai 0b1011, yang panjangnya 4 bit. Jadi, ekspresi (11).bit_length() akan bernilai 4.

3.

3.1. bin(i)

```
3  print(bin(i))
```

```
0b1000101011
```

Ekspresi bin(i) dalam Python mengembalikan string hasil biner dari bilangan bulat yang diberikan. Fungsi mengambil integer sebagai argumen dan mengembalikan string dalam format 0b diikuti dengan representasi biner dari integer. Dalam hal ini, i adalah variabel yang menyimpan nilai integer 555. Ekspresi bin(i) memanggil fungsi bin() pada i, yang mengembalikan string 0b1000100011, yang merupakan representasi biner dari 555.

3.2. hex(i)

```
5   print(hex(i))
```

```
0x22b
```

Fungsi bawaan hex() di Python mengembalikan representasi string hasil heksadesimal dari bilangan bulat pada parameter. Fungsi mengambil bilangan bulat sebagai argumen dan mengembalikan string dalam format 0x diikuti dengan representasi bilangan bulat heksadesimal. Dalam hal ini, i adalah variabel yang menyimpan nilai integer 555. Ekspresi hex(i) memanggil fungsi hex() pada i, yang mengembalikan string 0x22b, yang merupakan representasi heksadesimal dari 555.

3.3. oct(i)

```
7   print(oct(i))
```

```
0o1053
```

Fungsi built-in oct() di Python mengembalikan representasi string hasil oktal dari parameter yang diberikan. Fungsi mengambil integer sebagai argumen dan mengembalikan string dalam format 0o diikuti dengan representasi oktal dari integer. Dalam hal ini, i adalah variabel yang menyimpan nilai integer 555. Ekspresi oct(i) memanggil fungsi oct() pada i, yang mengembalikan string 0o1033, yang merupakan representasi oktal dari 555.

Kesimpulan:

Dalam Python, ada beberapa tipe data numerik yang dapat merepresentasikan angka, antara lain bilangan bulat (int), bilangan titik-mengambang (float), dan bilangan kompleks (kompleks).

- int mewakili bilangan bulat, yaitu bilangan bulat tanpa komponen pecahan. Misalnya, 10, -3, 0.
- float mewakili bilangan real dengan komponen pecahan. Misalnya, 3,14, -0,01, 1,0.
- kompleks mewakili bilangan kompleks, yang terdiri dari bagian nyata dan bagian imajiner. Bilangan kompleks ditulis dalam bentuk a + bj, dimana a adalah bagian real dan b adalah bagian imajiner. Misalnya, 3 + 4j, -1 + 0j.