Tutor Sebaya (2/5)
Pengenalan Komputasi

Stream Programming

Pengulangan

Made with love by Acads BPA STEI-K '23

Darrel Adinarya S. *19623199* *LINE: darrelas*
Z. Nayaka Athadiansyah *19623116* *LINE: nayaka.zna*

Sebelumnya...
Review Modul 1

Pengkom Cheatsheet: I/O dan Percabangan

Akademik BPA STEI-K 2023
Z. Nayaka Athadiansyah – 19623116
Darrel Adinarya S. – 19623199

Tipe Dasar

integer, float, boolean, string

```
  int   -196     0     165
float   16.68    0.11    5.0
 bool   True     False
  str   "STEI"    "Interupsi\nDanlap"
                  Menambahkan baris baru
```

Konversi Tipe

Mengkonversi nilai dari suatu *type* ke *type* lainnya. Sintaks umum: `type(ekspresi)`.

`int ("196")` → `196`

`float ("182.135")` → `182.135`

`str (2.3)` → `"2.3"`

Operator Aritmetika

Operator	Nama	Contoh	Hasil
+	Penjumlahan	`420 + 69`	489
-	Pengurangan	`2024 - 2023`	1
*	Perkalian	`49 * 4`	196
/	Pembagian (*float*)	`11 / 100`	0.11
//	Pembagian (*integer*, dibulatkan ke bawah)	`21 // 5`	4
%	Modulo (sisa bagi)	`12 % 5`	2
**	Perpangkatan	`5 ** 3`	125

Operator Relasional

Operator-operator berikut membandingkan dua bilangan lalu mengembalikan nilai *boolean*.

Operator	Nama	Contoh	Hasil
==	Sama dengan	`2.3 == 23/10`	True
!=	Tidak sama dengan	`11 != 11`	True
<, >, <=, >=	Membandingkan urutan dua bilangan (atau lebih)	`11 <= 11`	False

Operator Logika

Operator	Nama	Contoh
and	True jika keduanya True	True and False → False
or	True jika salah satu atau keduanya True	True or False → True
not	Membalikkan nilai kebenaran	not False → True

Operator *Assignment*

Tambahkan operator aritmetika sebelum tanda sama dengan (=) akan memodifikasi nilai variabel. Misalnya, `x += 2` akan menambahkan 2 ke nilai `x`.

F-String

Jika `nama = "Nayaka"` dan `sifat = "ganteng"` maka meng-*assign*

```
x = f"{nama} adalah orang yang {sifat}. "
    tambahkan f sebelum tanda kutip
```

berikabat `print(x)` memberikan output `"Nayaka adalah orang yang ganteng."`

Input/Output

- **Fungsi** `input()` menerima input berupa `string`.
- **Fungsi** `print()` mencetak output pada pengguna. Sintaks umumnya adalah
  `print(ekspresi, end='\n')`
  Secara *default* menambahkan baris baru; bisa diotak-atik.

Beberapa Cara Memakai `print()`

```python
# Cara 1
print("Balonku", "ada", "5.")
# Cara 2
print("Balonku" + "ada" + "5.")
# Cara 3
print("Balonku", end=" ")
print("ada", end=" ")
print("5.")
# Semuanya memberikan output "Balonku ada 5".
x = 5
print("Balonku ada {x}."}
# Output: "Balonku ada 5."
```

Percabangan

```python
# If statement
if <kondisi-1>:
    <aksi-1>

# Menggunakan else untuk melakukan aksi lain jika kondisi tidak terpenuhi
if <kondisi-1>:
    <aksi-1>
else: # <kondisi-1> tidak terpenuhi
    <aksi-2>

# Menggunakan elif untuk beberapa syarat tertentu
if <kondisi-1>:
    <aksi-1>
elif <kondisi-2>:
    <aksi-2># dan seterusnya
```

Materi

Perulangan (loop)

Loop: Buat Apa?

Loop dalam pemrograman digunakan untuk menjalankan serangkaian pernyataan secara berulang, memungkinkan penggunaan efisien dari kode untuk tugas yang memerlukan suatu aksi yang sama untuk dilakukan berkali-kali.

Contoh:

```
# Print huruf "a" sebanyak 100 kali
print('a')
print('a')
print('a')        vs     for i in range(100):
print('a')                    print('a')
… hingga 100 kali
print('a')
```

Kalian mending yang mana?

for loop

```
for i in range(n): # range(0,n)
    lakukan_sesuatu()
# atau
for i in range(*start, stop, *step):
    lakukan_sesuatu()
```

start : Indeks pertama (*optional: default value is 0)
stop : Indeks terakhir (tidak termasuk dalam intervalnya)
step : Beda antarsuku dalam barisan (*optional: default value is 1)

while loop

```
while (kondisi_terpenuhi):
    lakukan_sesuatu()
```

Kode dalam blok while akan dieksekusi berulang kali selagi kondisi masih terpenuhi.

For vs. While

For loop digunakan ketika kita sudah mengetahui jumlah pasti pengulangan yang ingin dilakukan

While loop digunakan ketika kita ingin melakukan pengulangan selama suatu kondisi terpenuhi

for loop dengan while loop

```
for i in range(2, 100):
    print(i)
```

```
i = 2
while (i < 100):
    print(i)
    i += 1
```

Bantai soal, yuk!

Problem 1: Dasar For Loop

Nayaka dan Darrel bermain **batu-gunting-kertas** selama *N* ronde (N > 0) untuk memutuskan siapa yang paling ganteng di antara mereka berdua. Agar pertandingan berlangsung secara adil, mereka memintamu menjadi jurinya.

Buatlah program yang menerima *N* dan string yang mewakili gerakan yang dikeluarkan tiap rondenya: “batu”, “gunting”, atau “kertas”. Program lalu memberikan skor hasil pertandingan (format: Nayaka-Darrel) dan menentukan pemenangnya, atau seri bila tidak ada yang menang. Perhatikan *test case* sebagai contoh dan asumsikan input valid.

[Diadaptasi dari Praktikum Pengenalan Komputasi STEI 2021]

Test Case 1:
Masukkan nilai N: 3
Gerakan Nayaka di ronde 1: batu
Gerakan Darrel di ronde 1: kertas
Gerakan Nayaka di ronde 2: gunting
Gerakan Darrel di ronde 2: gunting
Gerakan Nayaka di ronde 3: gunting
Gerakan Darrel di ronde 3: batu

Darrel menang dengan skor 0-2.

Test Case 2:
Masukkan nilai N: 2
Gerakan Nayaka di ronde 1: batu
Gerakan Darrel di ronde 1: gunting
Gerakan Nayaka di ronde 2: gunting
Gerakan Darrel di ronde 2: batu

Pertandingan seri dengan skor 1-1.

Test Case 3:
Masukkan nilai N: 1
Gerakan Nayaka di ronde 1: batu
Gerakan Darrel di ronde 1: gunting

Nayaka menang dengan skor 1-0.

Solusi Problem 1: Dasar For Loop

```python
# Input
n = int(input("Masukkan nilai N: "))

# Inisialisasi awal
nayScore = 0
relScore = 0

# Proses

for i in range(n):
    # Input gerakan
    nayMove = input(f"Gerakan Nayaka di ronde {i+1}: ")
    relMove = input(f"Gerakan Darrel di ronde {i+1}: ")

    # Menentukan perolehan skor pada ronde-(i + 1)
    # Nayaka menang
    if (nayMove == "batu" and relMove == "gunting") or (nayMove == "gunting" and relMove == "kertas") or
       (nayMove == "kertas" and relMove == "batu"):
        nayScore += 1
    elif (nayMove == relMove): # seri
        nayScore += 0
    else: # Darrel menang
        relScore += 1

if nayScore > relScore:
    print(f"Nayaka menang dengan skor {nayScore}-{relScore}.")
elif nayScore == relScore:
    print(f"Pertandingan seri dengan skor {nayScore}-{relScore}.")
else: # Darrel menang
    print(f"Darrel menang dengan skor {nayScore}-{relScore}.")
```

Problem 2:
Dasar While Loop

Sebuah bola awalnya dijatuhkan dari ketinggian ***h*** meter lalu memantul ke tanah. Ketinggian maksimum setelah tiap pantulan adalah ketinggian maksimum pantulan sebelumnya dibagi ***x***. Tentukan banyaknya pantulan yang terjadi hingga tinggi maksimum pantulan bola kurang dari 1 jika diberikan input bilangan real (*float*) **h** dan bilangan asli **x** dengan **h > 1**.

[Diadaptasi dari Praktikum Pengenalan Komputasi STEI 2022]

Test Case 1:
Ketinggian awal bola (h): 20
Nilai x: 2

Terjadi 5 pantulan.

Test Case 2:
Ketinggian awal bola (h): 20
Nilai x: 1

Pantulan terjadi selamanya.

Test Case 3:
Ketinggian awal bola (h): 69
Nilai x: 420

Terjadi 1 pantulan.

Solusi Problem 2: Dasar While Loop

```python
# Input
h = float(input("Ketinggian awal bola (h): "))
x = int(input("Nilai x: "))

# Inisialisasi Awal
count = 0

# Proses
if x == 1:
    print("Pantulan terjadi selamanya.")
else: #x > 1
    while (h >= 1):
        h /= x
        count += 1

    # Output
    print(f"Terjadi {count} pantulan.")
```

Problem 3: For Loop Bercabang

Ketika sedang mengerjakan soal latihan pemrograman, Nayaka menemukan sebuah pola unik segitiga istimewa, yakni sebuah segitiga siku-siku yang dibentuk dengan mengisi baris segitiga dengan angka berurutan, dimulai dengan angka 1 di sudut kiri atas. Buatlah sebuah program untuk membangun segitiga tersebut dengan tinggi t! Dipastikan t > 0.

[Diadaptasi dari Praktikum Pengenalan Komputasi STEI 2023]

```
Test Case 1: Test Case 2: Test Case 3:
Tinggi segitiga (t): 3 Tinggi segitiga (t): 5 Tinggi segitiga (t): 7

1
2 3
4 5 6

1
2 3
4 5 6
7 8 9 10
11 12 13 14 15

1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
16 17 18 19 20 21
22 23 24 25 26 27 28
```

Solusi 3: For Loop Bercabang

```
# Input
h = int(input("Tinggi segitiga (t): "))

# Pointer
a = 1

# Proses
for i in range(h+1):
    for j in range(i):
        print(a, end=" ")
        a += 1
    print()
```

Problem 4: Pola Bilangan

Darrel ingin membuat sebuah barisan bilangan yang dimulai dari angka 1. Barisan ini akan memiliki panjang sebanyak **x** bilangan. Namun, Darrel ingin barisan ini memiliki sifat unik, yaitu saat bertemu dengan kelipatan **y**, barisan tersebut akan mulai menurun dari kelipatan tersebut hingga mencapai angka **1**, dan setelah itu akan terus meningkat lagi sampai bertemu dengan kelipatan **y** selanjutnya.

Bantu Darrel membuat program untuk membentuk barisan bilangan tersebut.

[Diadaptasi dari Praktikum Pengenalan Komputasi shift 4.3 (STEI 2023)]

Test Case 1:
Masukkan x: 13
Masukkan y: 3

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

Test Case 2:
Masukkan x: 15
Masukkan y: 2

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

Test Case 3:
Masukkan x: 10
Masukkan y: 1

1 2 1 2 3 2 1 2 3 4

Solusi 4: Pola Bilangan

```
## Input
x = int(input("Masukkan x: "))
y = int(input("Masukkan y: "))

## Inisialisasi
count   = 0
yi      = y

## Proses
### While loop; dilakukan selagi panjang barisan
belum melebihi x
while count < x:
    i = 1
    #### Barisan Naik
    while (i < y and count < x):
        print(i, end=" ")
        i += 1
        count += 1
    # Setelah loop ini berakhir, i mencapai y
    #### Barisan Turun
    y += yi
    while (i > 1 and count < x):
        print(i, end=" ")
        i -= 1
        count += 1
```

Problem 5: Fibonacci

Barisan Fibonacci adalah barisan yang cukup terkenal dalam matematika. Barisan ini memiliki aturan sederhana: suatu suku dihasilkan dari penjumlahan dua suku sebelumnya. Barisan Fibonacci yang kali ini kita gunakan berbentuk seperti ini:

1, 2, 3, 5, 8, 13, 21, 34, ...

Diberikan bilangan bulat ***N***, Anda diminta untuk memberikan banyaknya suku barisan Fibonacci yang genap di bawah ***N***.

Test Case 1:
Masukkan N: 3
1

Test Case 2:
Masukkan N: 10
2

Test Case 3:
Masukkan N: 35
3

Test Case 4:
Masukkan N: 10000
6

Test Case 5:
Masukkan N: 2
0

Solusi Problem 5: Fibonacci

```python
# Input
n = int(input("Masukkan N: "))

# Inisialisasi Awal
fn1     = 1
fn2     = 2
count   = 0

# Proses
while (fn2 < n):
    # Counter ditambah ketika menemukan suku fibonacci yang genap
    if (fn2 % 2 == 0):
        count += 1
    # Beralih ke suku selanjutnya
    temp = fn1
    fn1 = fn2
    fn2 += temp

# Output
print(count)
```

Problem 6: Berurusan dengan Prima

Diberikan bilangan asli **N**, periksa apakah **N** prima. Sebagai bonus, buatlah juga program lain yang memberikan semua faktor N yang prima.

Test Case 1:	Test Case 2:	Test Case Bonus 1:	Test Case Bonus 2:
Masukkan N: 5	Masukkan N: 12	Masukkan N: 120	Masukkan N: 110
True	False	2 3 5	2 5 11

Solusi Problem 6: Berurusan dengan Prima

```
# Input
n = int(input("Masukkan N: "))

# Asumsi Bilangan Prima
prime = True

for i in range (2,n): # Untuk semua bilangan dari 2 hingga N-1
    if n % i == 0: # Apakah N dapat dibagi habis oleh bilangan tersebut?
        prime = False # Jika iya, N bukan merupakan bilangan prima

# Output Status Prima
print(prime)
```

Bonus Problem 6: Berurusan dengan Prima

```
# BONUS: Faktor dari N yang prima
# Input
n = int(input("Masukkan N: "))

# Proses
for i in range (2,n): # Untuk semua bilangan dari 2 hingga N-1
    if n % i == 0: # Apakah N dapat dibagi habis oleh bilangan tersebut?
        prime = True
        for j in range (2,i): # Apakah faktor tersebut merupakan bilangan prima?
            prime = False if i % j == 0 else prime
        print(i, end=" ") if prime else 0 # Output faktor tersebut apabila ia prima
```

Semangat!

Makasih udah mau belajar bareng malam ini.

Selamat beristirahat! :D