

Algoritma & Pemrograman

Pertemuan 6: Dekomposisi Fungsional
(Konsep Prosedur dan Fungsi)

Pendahuluan: Mengapa perlu ada prosedur dan fungsi?

- Semakin besar cakupan permasalahan yang akan diselesaikan, maka semakin banyak pula baris kode/perintah program yang harus ditulis
 - Contoh:
 - Windows XP memiliki 45 juta baris kode (wikipedia.com)
 - Linux kernel 2.6.35 memiliki 13.5 juta baris kode
- Program yang memiliki baris kode yang banyak perlu diatur agar:
 - Memudahkan kita (yakni programmer) untuk menyelesaikan permasalahan (yakni dengan menulis program) secara sistematis → mulai dari hal yang abstrak lalu menuju ke hal yang lebih detail
 - Memudahkan kita untuk menelusuri program jika terdapat *bug*
 - Memudahkan kita atau programmer lain untuk menambahkan baris kode lain sebagai bentuk pengembangan aplikasi

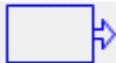
Abstraksi dan Prosedur

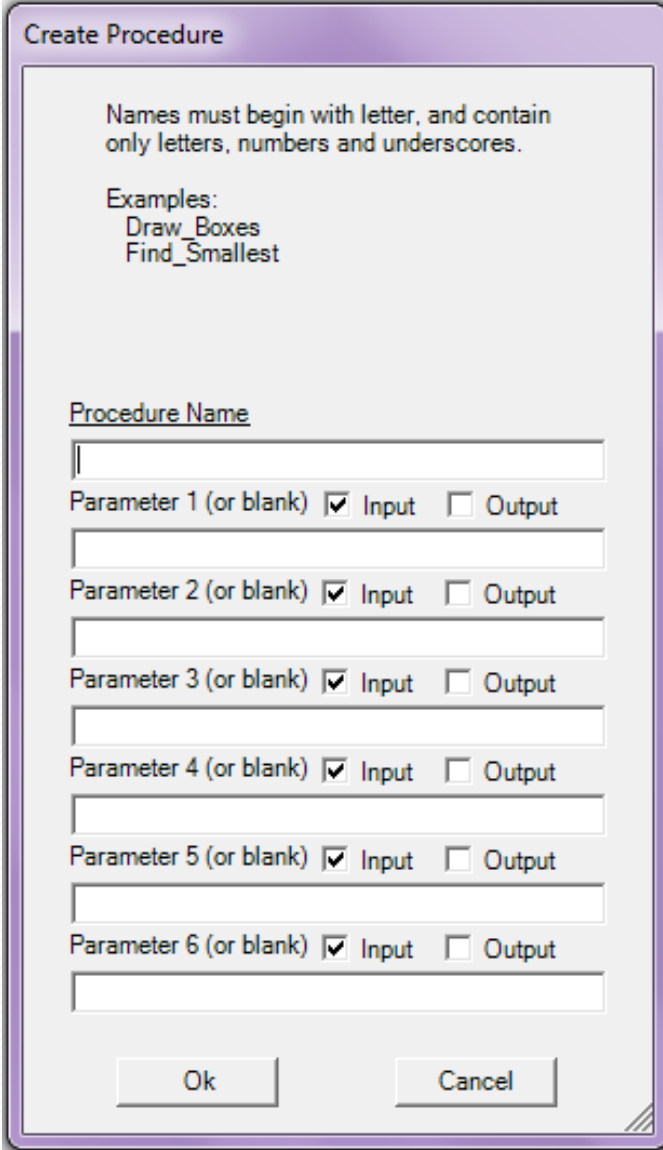
- Pada bidang ilmu komputer, **abstraksi** adalah elemen kunci untuk penyelesaian masalah
- Untuk menyelesaikan sebuah permasalahan kompleks, kita perlu berpikir secara garis besar tanpa memikirkan semua detail yang terkait (setidaknya pada awalnya)
- Dalam pemrograman, kita meng-abstraksi-kan detail dengan mengelompokkan serangkaian instruksi/baris program ke dalam unit-unit yang disebut **prosedur**

Prosedur/Fungsi/Method

- Merupakan pengelompokan beberapa instruksi/baris program yang melakukan sebuah perhitungan/komputasi tertentu
- Prosedur dapat menerima **parameter input** dan menghasilkan/mengembalikan **parameter output**
 - **Parameter input** : variabel yang diberikan pada prosedur untuk diproses di dalam prosedur
 - **Parameter output** : disebut juga nilai kembalian (*return value*), yakni variabel yang merupakan “output” dari prosedur

Membuat prosedur di Raptor

- Pastikan “Mode” yang aktif adalah **Intermediate**
- Klik kanan pada tab “main” dan pilih “Add procedure”
- Raptor dapat menerima maksimal 6 parameter untuk sebuah prosedur
- Sebuah parameter dapat menjadi parameter input, parameter output, atau parameter input dan output
- Prosedur yang telah dibuat lalu dapat dipanggil dari “main” dengan menggunakan simbol “Call” 



Create Procedure

Names must begin with letter, and contain only letters, numbers and underscores.

Examples:
Draw_Boxes
Find_Smallest

Procedure Name

Parameter 1 (or blank) ☒ Input ☐ Output

Parameter 2 (or blank) ☒ Input ☐ Output

Parameter 3 (or blank) ☒ Input ☐ Output

Parameter 4 (or blank) ☒ Input ☐ Output

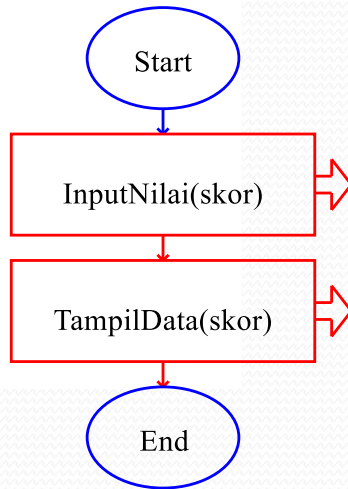
Parameter 5 (or blank) ☒ Input ☐ Output

Parameter 6 (or blank) ☒ Input ☐ Output

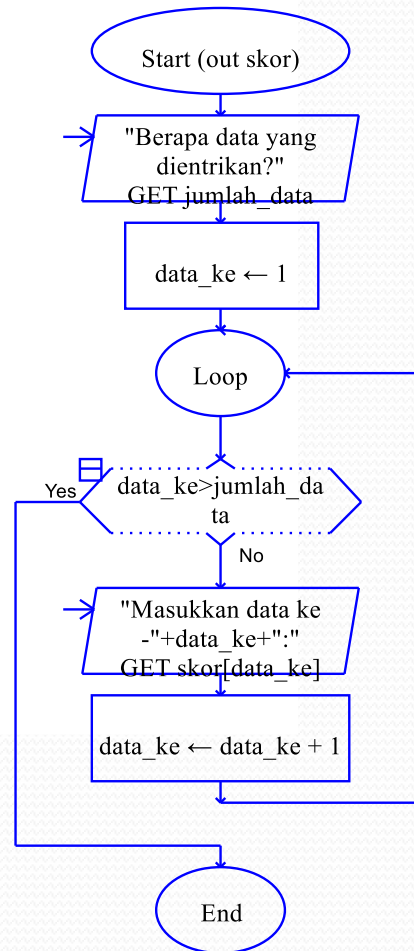
Ok Cancel

Contoh prosedur sederhana

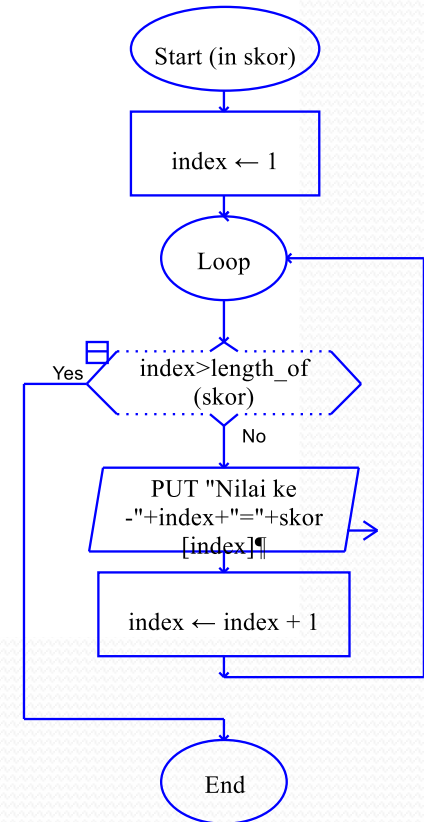
Main



Prosedur InputNilai



Prosedur TampilData



Previous example: Hitung nilai IPK

- Kita tinjau ulang algoritma untuk menghitung nilai IPK pada contoh sebelumnya.
- Abstraksi algoritma untuk menyelesaikan masalah tersebut adalah:
 - Meminta input nilai dan bobot SKS dari user
 - Mengkonversi nilai huruf ke nilai angka
 - Menghitung perkalian nilai angka dengan bobot SKS dan tambahkan ke jumlah dari perkalian nilai angka dengan bobot SKS $\rightarrow X$
 - Menghitung jumlah dari total SKS $\rightarrow Y$
 - Menghitung nilai IPK sesuai rumus: X/Y
 - Menampilkan hasil perhitungan nilai IPK ke layar

Hitung nilai IPK (lanjutan)

- Contoh2 yang telah diberikan di pertemuan sebelumnya dapat diperbaiki agar lebih mudah dibaca dan ditelusuri
- File: test5-prosedur.rap

Latihan

- Perbaiki lagi contoh di `test5-prosedur.rap` dengan menjadikan proses konversi nilai huruf ke angka ke prosedur terpisah
 - **Petunjuk:**
 - Untuk prosedur konversi nilai:
 - Parameter input: nilai (dalam bentuk huruf)
 - Parameter output: nilai (dalam bentuk angka)
 - Output prosedur yang sudah berupa angka, akan dijadikan salah satu parameter input bagi prosedur `Hitung_Total_Nilai_x_SKS`

Lanjutan latihan: Menghitung Standar Deviasi

- Standar deviasi adalah cara mengukur persebaran data. Semakin besar persebaran data terjadi, semakin besar nilai dari standar deviasi.
 - Contoh: terdapat 2 hasil ujian akhir dari sebuah kelas dengan 30 siswa. Hasil I: nilai bervariasi mulai 31-98, dan Hasil II: nilai bervariasi mulai 82-93. Nilai standar deviasi dari Hasil I akan lebih besar dari standar deviasi pada Hasil II.
- Buat flowchart untuk menghitung standar deviasi dari sekumpulan data yang diinputkan oleh user.

• Rumus Standar Deviasi, σ :

- Dimana:

- Σ adalah simbol penjumlahan deret
- x adalah data, x_i berarti data ke- i
- x_{avg} adalah rata-rata dari seluruh data
- n adalah banyaknya data

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - x_{avg})^2}{n - 1}}$$

- Gunakan prosedur/fungsi dalam membuat flowchart yang mencari standar deviasi dari sekumpulan data yang diinputkan oleh user

Petunjuk:

- Anda dapat menentukan sendiri bagian mana yang perlu dibuatkan prosedur/fungsi terpisah
- Anda dapat menggunakan fungsi `sqrt(x)` pada Raptor untuk mencari akar kuadrat dari x