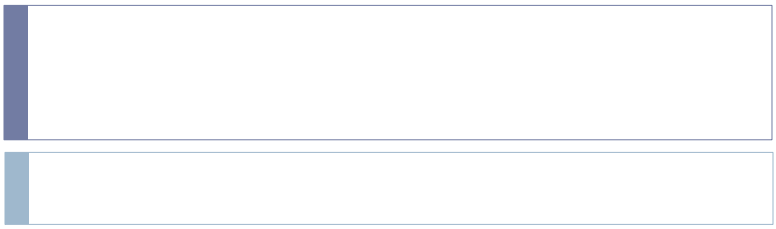
Praktikum 2
Bahasa Pemrograman

PLT Scheme : Membuat Program



Latihan 1

Diberikan tiga bilangan bulat. Program akan memeriksa apakah bilangan yang di tengah merupakan hasil kali bilangan di kiri dan kanannya.

```
> (cek 2 5 9)
#f
> (cek 3 15 5)
#t
```

- Contoh: fungsi faktorial (n)
  - Jika n = 0, nilainya 1
  - Selainnya, nilainya n * faktorial (n – 1)

```
> (faktorial 3)
  6
```

Latihan

Menghitung banyaknya bilangan bulat pada selang [a, b] yang habis dibagi oleh n. Contoh:

```
> (hitung 3 20 4)
5
```

```
; Menghitung banyaknya
; bilangan antara 3 s/d 20 yg
; habis dibagi 4 Ada 5 :
;4, 8, 12, 16, 20
```

Latihan 3

Menghitung banyaknya faktor bilangan bulat selain 1 dan bilangan itu sendiri.
Contoh:

```
> (faktor 15)
2                ; Ada dua faktor, yaitu 3 dan 5
                 ; Bilangan yang habis membagi bilangan
                 ; tersebut
```

Latihan 4

- Buatlah Program dengan Scheme untuk menentukan bilangan fibonacci

Latihan 5

FIRST and REST

- FIRST returns the first element of the list

```
* (first '(fast computer are nice))
* (first '(a b c))
* (first '((a b) (c d)))
 * (first '(a))
```

- REST does the complementary thing. It returns a list containing all but the first element

```
* (rest '(fast computer are nice))
* (rest '(a b c))
 * (rest '((a b)(c d))
```

- FIRST dan REST pada list kosong ( ) → error, expected argument of type <non-empty list>

CAR and CDR

- Sama seperti FIRST dan REST
- Memungkinkan menggunakan CXXR, CXXXR atau CXXXXR ; dimana setiap X dapat berupa A (menjadi CAR) atau D (menjadi CDR)

```
* (cadr '(a b c)) ≡
  (car (cdr '(a b c))) ≡
  (first (rest '(a b c)))
```

- (cadr <exspression>)
  (car (cdr <expression>))

Quoting Stop Evaluation

- Misalkan akan mengambil elemen kedua dari list (A **B** C)

```
* (first (rest (a b c))) → error
* (first (rest '(a b c)))
* (first '(rest (a b c)))
```

→ Quoting digunakan sebagai batas pembeda antara prosedur dengan argumen

CONS and LIST (1)

- CONS and LIST contruct List
- CONS (CONStruct) takes an **expression** and a **list** and returns a new list whose first element is the expression and whose remaining elements are those of the old list

(CONS <new first element> <a list>)

```
* (cons 'a '(b c))

> (define new 'a)
> (define old '(b c))
> (cons new old)
  (a b c)
> (first (cons new old))
  a
> (rest (cons new old))
  (b c)
```

CONS and LIST (2)

- LIST : makes a list out of its arguments (each element becomes an element of the new list)

```
> (list 'a 'b 'c)
  (a b c)
> (list 'a '(b c) 'd)
  (a (b c) d)
> (list '(a b) '())
  ((a b) ())
```


REMOVE

- REMOVE : removes an element from a list

```
* (remove `algor `(basprog algor orkom))
> (remove 'a '(a b c))
  (b c)
> (remove 'a '(makan malam))
  (makan malam)
```

LENGTH and REVERSE

- **LENGTH** primitive counts the number of top-level elements in a **list**.

```
> (length '()) → 0
> (length '(a b c d)) → 4
> (length '(() a b c)) → 4
> (length '((plato socrates atistotle))) → 1
> (length ())
> (length 'a)
```

- **REVERSE** reverses the order of the top-level elements of a list.

```
* (reverse '((plato socrates aristotle)))
> (reverse 'a)
> (reverse '(a b c)) → (c b a)
> (reverse '(a (b c) d)) → (d (b c) a)
> (reverse '(a (b c) d e)) → (e d (b c) a)
```


DATA TYPES (1)

- Number
  - Integers : represent whole numbers
    e.g. 47
  - Ratio : represent rational number
    e.g. 4/7
  - Floating point : represent real
    e.g. 4.7

DATA TYPES (2)

- The result of division depends on the number involved
  - floating / floating = floating
    e.g. `* (/ 1.234321 1.111)`
  - int / int =
    - **integer** when given two integer arguments that happen to divide evenly
    - **ratio** when given two integer arguments that do not divide evenly

* `(/ 1 2)` ≡ `(/ 2)`
* `(- 8)` and `(- -8)`

Few primitives for numbers

- MIN and MAX
  * `(max 2 4 3)`
  * `(min 2 4 3)`
  * `(* (max 3 4 5) (min 3 4 5))`
  * `(min (max 3 1 4) (max 2 7 1))`
- EXPT calculates powers, it raises its argument to its second
  * `(expt 2 3)`
  * `(expt 3 2)`
- SQRT takes the square root
  * `(sqrt 9)`
- ABS computes the absolute value
  * `(abs 5)`
  * `(abs -5)`

Exercise1: Problem FIRST and REST

- Evaluate the following form
  - `(first '((a b) (c d)))`
  - `(rest '((a b) (c d)))`
  - `(first (rest '((a b) (c d))))`
  - `(rest (first '((a b) (c d))))`
  - `(first (rest (first '((a b) (c d)))))`
  - `(first (first (rest (rest '((a b) (c d) (e f))))))`
  - `(first (first (rest '(rest ((a b) (c d) (e f))))))`
  - `(first (first '(rest (rest ((a b) (c d) (e f))))))`
- Write sequences of FIRST and REST that will pick the symbol PEAR out of the following expression

```
(apple orange pear grapefruit)
((apple orange) (pear grapefruit))
(((apple) (orange) (pear) (grapefruit)))
(apple (orange) ((pear)) (((grapefruit))))
 ((((apple))) ((orange)) (pear) grapefruit)
 ((((apple) orange) pear) grapefruit)
```


Exercise2

- Tugas akan saya berikan lewat blog
- Dapat dilihat setelah kegiatan praktikum selesai.

www.totoharyanto.staff.ipb.ac.id