テキスト処理 第11回 (2007-07-17)
e+?, e{m,n}? レポート説明

田中哲
産業技術総合研究所
情報技術研究部門
akr@isc.senshu-u.ac.jp
http://staff.aist.go.jp/tanaka-akira/textprocess-2007/

レポート

- 以下を実装して解説せよ
  - e+?
  - e{m,n}?
- 実装したらユニットテストで確認すること
- ∠切 2007-07-17 16:20
- HIPLUS
- 拡張子が txt なテキストファイル希望

/e+?/

- /e+/ の lazy 版
- 抽象構文木では [:plus_lazy, e]

- matchstr([:plus_lazy, [:lit, "a"]], "aaaaa")
  #=> [1,2,3,4,5]

/e{m,n}?/

- /e{m,n}/ の lazy 版
- 抽象構文木では [:times_lazy, e, m, n]

- matchstr([:times_lazy, [:lit, "a"], 2, 4], "aaaaa")
  #=> [2,3,4]

/e+?/ の実装 (1)

```
def try(exp, seq, pos, &block)
  ...
  when :plus_lazy
    _, e = exp
    try_plus_lazy(e, seq, pos, &block)
  ...
end
```

/e+?/ の実装 (2)

```
def try_plus_lazy(e, seq, pos, &block)
  try(e, seq, pos) {|pos2|
    try_rep_lazy(e, seq, pos2, &block)
  }
end
```

try_plus との比較

```
def try_plus_lazy(e, seq, pos, &block)
  try(e, seq, pos) {|pos2|
    try_rep_lazy(e, seq, pos2, &block)
  }
end
def try_plus(e, seq, pos, &block)
  try(e, seq, pos) {|pos2|
    try_rep(e, seq, pos2, &block)
  }
end
```

1回 try で成功した後、
呼び出すメソッドが違う

/e{m,n}?/ の実装 (1)

```
def try(exp, seq, pos, &block)
  ...
  when :times_lazy
    _, e, m, n = exp
    try_times_lazy(e, m, n, seq, pos, &block)
  ...
end
```

/e{m,n}?/ の実装 (2)

```
def try_times_lazy(e, m, n, seq, pos, &block)
  yield pos if m <= 0
  if 0 < n
    try(e, seq, pos) {|pos2|
      try_times_lazy(e, m-1, n-1, seq, pos2, &block)
    }
  end
end
```

try_times との比較

```
def try_times_lazy(e, m, n,
    seq, pos, &block)
  yield pos if m <= 0
  if 0 < n
    try(e, seq, pos) {|pos2|
      try_times_lazy(e, m-1, n-1,
        seq, pos2, &block)
    }
  end
end
```

```
def try_times(e, m, n,
    seq, pos, &block)
  if 0 < n
    try(e, seq, pos) {|pos2|
      try_times(e, m-1, n-1,
        seq, pos2, &block)
    }
  end
  yield pos if m <= 0
end
```

先に yield するか、後に yield するかが違う

ざっと眺めた結果

- 動かないものを提出したひとは見当たらなかった
- ちょっと無駄な動作をする実装があった
  - try_times_lazy で n = 0 のときに try を再帰的に呼び出すことは不要