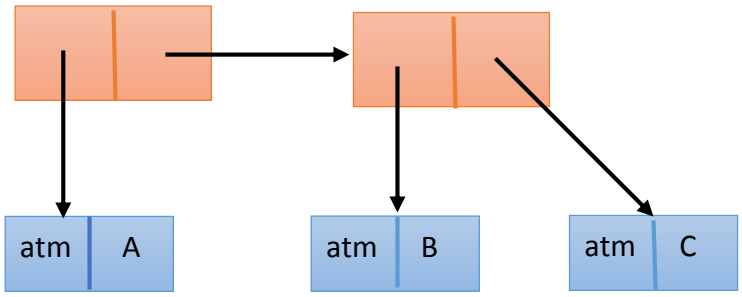
Programing language – homework 4

مرجان آلبویه ۹۱۳۱۰۶۰

3.1:

a)



b)

(cons ((cons 'A (cons 'B 'C)) (cons 'B 'C)))

در cons اولی یک cons cell ایجاد میشود که قسمت car آن به cons cell ای اشاره دارد که قسمت car آن A است و قسمت cdr آن خانه ای دیگری است که car و cdr آن به ترتیب به B و C اشاره دارند. در نهایت قسمت decrement خانه اولی به خانه ای دیگر اشاره دارد که قسمت car آن B و cdr آن به C اشاره میکند.

c)

((lambda (x)(cons (cons 'A x)x))(cons 'B 'C))

ابتدا (cons 'B 'C) را evaluate میکنیم و سپس آن را روی تابع apply میکنیم.

3.2:

a)

امکان پذیر نیست زیرا اگر P_i تعریف نشده باشد برنامه متوقف میشود یعنی halt اتفاق می افتد زیرا شرط ها به صورت ترتیبی اجرا میشوند و با رسیدن به یک شرط تعریف نشده برنامه متوقف میشود.

b)

در این روش شرط ها به صورت موازی اجرا میشوند و در صورت رخ دادن halt متوقف میشود. از بین شرط هایی که به درست هستند مقدار یکی انتخاب میشود (لزومی ندارد که مقدار اولین شرط درست باشد)

c)

```
(defun odd (x) (cond ((eq x 0) nil)
                     ((eq x 1) t)
                     ((> x 1) (odd (- x 2)))
                     ((< x 0) (odd (+ x 2)))))
```

d)

برای Scor روش a بهتر است زیرا به صورت ترتیبی اجرا میشود یعنی اگر شرط اول برقرار نبود به سراغ شرط دوم میرود.

برای Por روش b بهتر است زیرا شرط ها به صورت موازی اجرا میشوند هرکدام درست باشد نتیجه مقدار همان است البته در این روش otherwise باید کامل تعریف شده باشد.

اگر برای Scor روش b را به کار ببریم بعضی شرط ها دوبار چک میشوند یعنی کار اضافی انجام میشود و دیرتر به جواب میرسیم و اگر برای Por روش a اجرا شود مناسب نیست زیرا برای Por شرط ها میتوانند موازی اجرا شوند بدون اینکه به شرط های دیگر کار داشته باشند که اینکار سریع تر است.

3.4:

a)

```
f (h xs)
```

b)

```
g=maplist
h=car
```

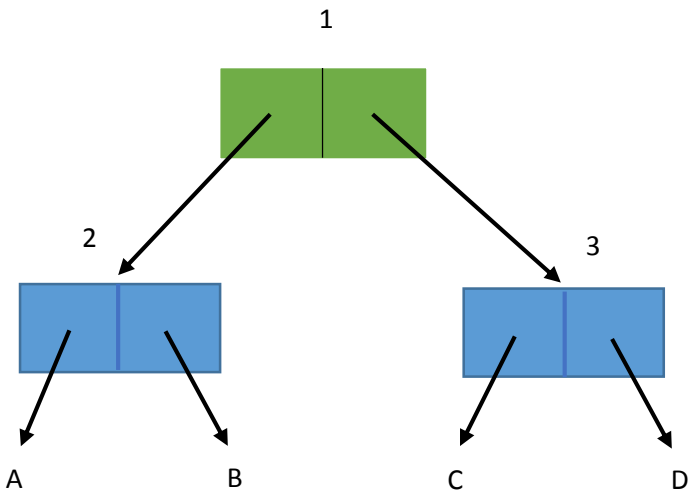
c)

compose (f h)

3.6:

a)

نتیجه این عبارت اتم c است . و cell های شماره ۱ و ۲ و ۳ به عنوان garbage دور ریخته میشوند.



b)

مثلا عبارت زیر را cdr(rplacd (xx)) در نظر بگیرید. شماره گر x یکی کم میشود در صورتیکه نباید کم شود زیرا cdr دوباره به x اشاره میکند.

3.8:

a)

T(n)=Max(T(n-1) , T(n-2))

T(n)= ??

b)

(Lambda (e) (cons (car e) (rplaca e c))) (cons a b)

(Lambda (e) (cons (car future(e)) (rplaca e c))) (cons a b)

در عبارت اول دستورات به صورت پشت سرهم اجرا میشوند یعنی car e برابر a است و سپس در دستور rplaca e c قسمت آدرس عبارت e تغییر کرده و برابر c میشود

اما در عبارت دوم چون دستوری که شامل future است به موازات خود future process اجرا میشود بنابراین اگر rplaca e c اجرا شود (car future(e)) اینبار به c اشاره خواهد نه به a !!

c)

(cond (e1 p1) (e2 p2))

فرض کنید چنین عبارتی را داریم ابتدا e1 را بررسی میشود و فرض کنیم نادرست باشد سپس e2 بررسی میشود و فرض کنیم درست است بنابراین دستور p2 اجرا خواهد شد.حال اگر برنامه به این صورت تغییر کند:

(cond (future(e1) p1) (e2 p2))

در این حالت به صورت موازی هم future اجرا شده و هم e2 بررسی میشود چون e2 درست است پس باز هم p2 اجرا میشود در صورتی که ممکن بود نتیجه future(e1) درست باشد.

d)

به علت وجود concurrency ، errorها به صورت موازی اجرا میشوند اما ترتیب اجرای آن ها مشخص نیست و اگر برنامه ما هنگام اجرا چند خطا داشته باشد یکی از error handler ها به صورت رندم اجرا میشود و اگر برنامه را دوباره اجرا کنیم ممکن است handler دیگری اجرا شود یعنی یک برنامه در چند اجرا نتایج متفاوتی خواهد داشت.