طراحی زبان‌های برنامه‌سازی

پاییز ۱۴۰۰

استاد: محمد ایزدی

گردآورندگان: پارسا محمدیان، آرین یزدانپرست

بررسی و بازبینی: امیرعلی ابراهیم زاده

دانشگاه صنعتی شریف

دانشکده‌ی مهندسی کامپیوتر

تمرین چهارم — عبارات (فصل سوم کتاب) — مهلت ارسال: ۳۰ اردیبهشت ۱۴۰۱

به موارد زیر توجه کنید:

۱. برنامه‌های خود را به زبان Racket بنویسید.

۲. برنامه‌های شما برای این تمرین نیاز نیست قابل اجرا باشند.

۳. مهلت ارسال تمرین ساعت ۵۹ : ۲۳ روز جمعه ۳۰ اردیبهشت ۱۴۰۱ است.

۴. در نهایت تمام فایل‌ها را در یک فایل زیپ قرار داده و با نام $HW4\_StudentID$ در سامانه کوئرا آپلود کنید.

۵. هرگونه سوالی راجع به تمرین را در زیر پست مربوطه در کوئرای درس مطرح کنید.

۶. در مجموع تمامی تمارین ۷ روز مهلت تاخیر مجاز دارید و پس از تمام شدن این تاخیرهای مجاز به ازای هر روز ۱۰ درصد از کل نمره تمرین شما کم می‌شود.

۷. لطفا تمرین‌ها را از یکدیگر کپی نکنید. در صورت وقوع چنین مواردی مطابق با سیاست درس رفتار می‌شود.

سوالات (۱۰۰ نمره)

۱. (۱۵ نمره) برای عبارت زیر یک derivation tree رسم کنید. (نمونه‌ای از درخت استخراج را در صفحه ۵ کتاب می‌توانید مشاهده کنید)

Let ρ = [a=⌈45⌉, b=⌈5⌉].
(value-of
«let x = -(a, -(0, b)) in let y = -(a, b)
in if zero?(-(x, 100)) then -(y, 0) else -(x, 0)»
ρ)

۲. (۱۵ نمره) به زبان let تابع فاکتوریل را اضافه کنید. (تمامی قسمت‌های تغییر یافته در کد را مشخص کنید)

۳. (۲۰ نمره) **(الف)** زبان let را به طوری تغییر دهید که تابع let امکان اینکه چند ورودی بگیرد را داشته باشد. به عنوان مثال عبارت زیر خروجی ۵ را تولید می‌کند.

```
let x = 10 y = 5
in -(x,y)

```

(ب) همین امکان را به زبان letrec نیز اضافه کنید.

۴. (۲۵ نمره) **(الف)** تعیین کنید خروجی قطعه کد زیر برابر چه عددی است. نحوه رسیدن تابع به خروجی توسط این تابع را تحلیل کنید.

```
let maketimes = proc (maker)
                  proc (x)
                    proc (y)
                      if zero?(x)
                      then 0
                      else -((((maker maker) -(x, 1)) y), -(0, y))
in let times = (maketimes maketimes)
   in let f = proc (maker)
                       proc (x)
                         if zero?(x)
                         then 1
                         else ((times x) ((maker maker) -(x, 1)))
      in ((f f) 5)
```

(ب) با استفاده از روش بالا، برنامه‌ای در زبان proc بنویسید که اول بودن یک عدد صحیح مثبت را بررسی کند. در صورت اول بودن `#t` و در غیر این صورت `#f` برگرداند

۵. (۲۵ نمره) در زبان proc هنگام ذخیره یک procedure جدید، تمام environment را ذخیره می‌کنیم. در حالی که تنها به مقدار متغیرهای آزاد (free variables) نیاز داریم. حال تعریف proc-exp را به گونه‌ای تغییر دهید که تنها متغیرهای آزاد را در environment ذخیره کند. (راهنمایی: یک تابع برای تشخیص متغییرهای آزاد در یک عبارت بنویسید، سپس متغییرهای bounded را با استفاده از آن از محیط حذف کنید)