دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

طراحی زبان‌های برنامه‌سازی

بهار ۱۴۰۱

استاد: محمد ایزدی
گردآورندگان: سینا الهی منش، سروش جهان‌زاد، آرین احدی‌نیا

تمرین دوم | **برنامه‌نویسی تابعی** | مهلت ارسال: ۲۳ فروردین ۱۴۰۱

به موارد زیر توجه کنید:

۱) برنامه‌های خود را به زبان Racket بنویسید.
۲) مهلت ارسال تمرین ساعت ۵۹ : ۲۳ روز ۲۳ فروردین ۱۴۰۱ است.
۳) در مسائلی که نیاز به خروجی دادن است، پاسخ را چاپ نکنید بلکه آن را به عنوان حاصل تابع $main$ برگردانید.
۴) حتما نام تابع اصلی خود را $main$ بگذارید تا هنگام داوری مشکلی پیش نیاید. منظور از تابع اصلی تابعی است که اجرای آن روی ورودی، خروجی مطلوب را نشان می‌دهد.
۵) جواب هر سوال برنامه نویسی تک‌بخشی را در یک فایل Racket با نام و فرمت زیر قرار دهید.

$$[StudentID]\_[ProblemNumber].rkt$$

۵) جواب هر بخش از سوال‌های برنامه نویسی چندبخشی را در یک فایل Racket با نام و فرمت زیر قرار دهید.

$$[StudentID]\_[ProblemNumber]\_[PartNumber].rkt$$

۷) در نهایت فایل PDF پاسخ سوال اول و تمام فایل‌های Racket را در یک فایل زیپ قرار داده و با نام $HW۲\_StudentID$ در سامانه کوئرا آپلود کنید.
۸) هرگونه سوالی راجع به تمرین را در زیر پست مربوطه در کوئرای درس مطرح کنید.
۹) 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.
۱۰) لطفا تمرین‌ها را از یکدیگر کپی نکنید. در صورت وقوع چنین مواردی مطابق با سیاست درس رفتار می‌شود.

سوالات (۲۰ + ۱۰۰ نمره)

۱. (۲۰ نمره) در هر قسمت بنویسید که چه مجموعه‌هایی بر اساس این قواعد ساخته می‌شوند؟

A. $(0,1) \in S \quad \frac{(n,k)\in S}{(n+1,k+7)\in S}$

B. $(0,1) \in S \quad \frac{(n,k)\in S}{(n+1,2k)\in S}$

C. $(0,0,1) \in S \quad \frac{(n,i,j)\in S}{(n+1,j,i+j)\in S}$

D. $(0,1,0) \in S \quad \frac{(n,i,j)\in S}{(n+1,i+2,i+j)\in S}$

۲. (۲۰ نمره)

(الف) (۵ نمره) تابعی بنویسید که دو بردار هم‌اندازه را به صورت دو لیست بگیرد و ضرب داخلی (dot product) آن دو را خروجی بدهد.

فراخوانی نمونه

```
(main '(1 2 3 4 5) '(6 7 8 9 10))
```

خروجی نمونه

```
130
```

(ب) (۵ نمره) تابعی بنویسید که دو بردار هم‌اندازه را به صورت دو لیست بگیرد و ضرب خارجی (outer product) آن دو را خروجی بدهد.

فراخوانی نمونه

```
(main '(1 2 3 4 5) '(6 7 8 9 10))
```

خروجی نمونه

```
'((6 7 8 9 10) (12 14 16 18 20) (18 21 24 27 30) (24 28 32 36 40) (30 35
    40 45 50))
```

(پ) (۱۰ نمره) تابعی بنویسید که دو ماتریس را دریافت کند و حاصل ضرب اولی در دومی را خروجی بدهد. هر ماتریس به صورت لیستی از لیست‌ها (سطرها) نمایش داده می‌شود. فرض کنید که ابعاد ورودی‌های این تابع همواره به گونه‌ای هستند که ضرب آن‌ها امکان‌پذیر است.

فراخوانی نمونه

```
(main '((1 1) (1 0)) '((1 2) (3 4)))
```

خروجی نمونه

```
'((4 6) (1 2))
```

۳. (۲۰ نمره) تابعی بنویسید که دو عنصر a و b و یک لیست را در ورودی دریافت کند و تمام عناصری که در این لیست برابر a هستند را تبدیل به b و تمام عناصر که در لیست برابر b هستند را تبدیل به a کند. (به عبارتی a ها را با b و b ها را با a جایگزین کند.)

فراخوانی نمونه

```
(main 'a 'd '(a b c d))
```

خروجی نمونه

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

۴. (۲۰ نمره) سیستم رمزنگاری‌ای را در نظر بگیرید که دو پارامتر a و b دارد و روی کلماتی با حروف کوچک انگلیسی اعمال می‌شود. در این رمزنگاری، به ازای هر حرف، جایگاهش در الفبا در نظر گرفته می‌شود (0 برای a تا 25 برای z) و حرف x ام الفبا با حرف $((a \times x) + b \mod 26)$ ام جایگزین می‌شود. فاصله‌ها در این رمزنگاری دست‌نخورده باقی می‌مانند.

(الف) (۱۰ نمره) تابعی بنویسید که یک رشته شامل یک کلمه از حروف کوچک انگلیسی را به همراه a و b دریافت کند و رشته‌ای که پس از رمزنگاری به این شیوه به دست می‌آید را برگرداند.

فراخوانی نمونه

```
(main "hello world" 17 20)
```

خروجی نمونه

```
"jkzzy eyxzt"
```

(ب) (۱۰ نمره) تابعی بنویسید که یک رشته حاصل از رمزنگاری به این شیوه را به همراه a و b متناظرش دریافت کند و رشته‌ی اصلی پیش از رمزنگاری را برگرداند.

فراخوانی نمونه

```
(main "jkzzy eyxzt" 17 20)
```

خروجی نمونه

```
"hello world"
```

۵. (۲۰ نمره) تابعی بنویسید که عدد صحیح n را به عنوان ورودی بگیرد و سطر n ام مثلث خیام‌ـپاسکال را در یک لیست به عنوان خروجی برگرداند.

فراخوانی نمونه

```
(main 10)
```

خروجی نمونه

```
'(1 9 36 84 126 126 84 36 9 1)
```

۶. (۲۰ نمره) [امتیازی] مرتب‌سازی سریع یک روش مرتب‌سازی است که در سال ۱۹۵۹ توسط Tony Hoare معرفی شد. این روش برای مرتب‌سازی یک لیست ابتدا یک عنصر را به عنوان محور (pivot) انتخاب می‌کند. سپس عناصر کوچک‌تر از محور را در یک لیست و عناصر بزرگ‌تر از محور را در یک لیست جدا قرار میدهد و به صورت بازگشتی آنها را مرتب می‌کند. سپس با به هم چسباندن این دو لیست و عنصر محوری، لیست مرتب‌شده را خروجی می‌دهد.

برای انتخاب محور میتوان به طریق دلخواه عمل کرد. مثلا میتوان عنصر اول یا آخر لیست را به عنوان محور در نظر گرفت. اما برای جلوگیری از حملات خصمانه، خوب است که به صورت random این عنصر انتخاب شود. در این پیاده‌سازی نیز شما ملزم هستید تا به صورت تصادفی با احتمال یکنواخت (uniform) یکی از عناصر لیست را به عنوان محور انتخاب کنید.

در این تمرین شما باید تابعی به نام quick-sort بنویسید که با استفاده از الگوریتم مرتب‌سازی سریع لیستی از عناصر را مرتب کند. این تابع دو ورودی دارد. ورودی اول comparator است که خود یک تابع است و نتیجه مقایسه دو عنصر را برمیگرداند. ورودی دوم یک لیست است که باید برمبنای مقایسه‌گر داده‌شده مرتب شود.

تابع comparator تابعی است که دو ورودی دریافت می‌کند و برمبنای زیر خروجی می‌دهد.

- اگر ورودی اول از ورودی دوم کوچک‌تر باشد، باید یک عدد منفی برگردانده شود.

- اگر ورودی اول با ورودی دوم برابر باشد، باید صفر برگردانده شود.
- اگر ورودی اول از ورودی دوم بزرگ‌تر باشد، باید یک عدد مثبت برگردانده شود.

جهت شفاف‌سازی، یک لیست مرتب بر مبنای یک مقایسه‌گر، لیستی است که اگر دو عنصر دلخواه مانند x و y را در آن انتخاب کنیم به نحوی که x سمت چپ y باشد، آنگاه حاصل (comparator x y) مقداری نامثبت خواهد بود.

تضمین میشود که تابع comparator خاصیت تعدی و پادمتقارن بودن را داشته باشد.

فراخوانی نمونه ۱

```
(quick-sort (lambda (x y) (if (= x y) 0 (if (< x y) 1 -1))) '(1 6 4 3 2
   5))
```

خروجی نمونه ۱

```
'(6 5 4 3 2 1)
```

فراخوانی نمونه ۲

```
(quick-sort - '(1 6 5 4 3 2))
```

خروجی نمونه ۲

```
'(1 2 3 4 5 6)
```

موفق باشید! (:−