Homework 0 - Basic Scheme

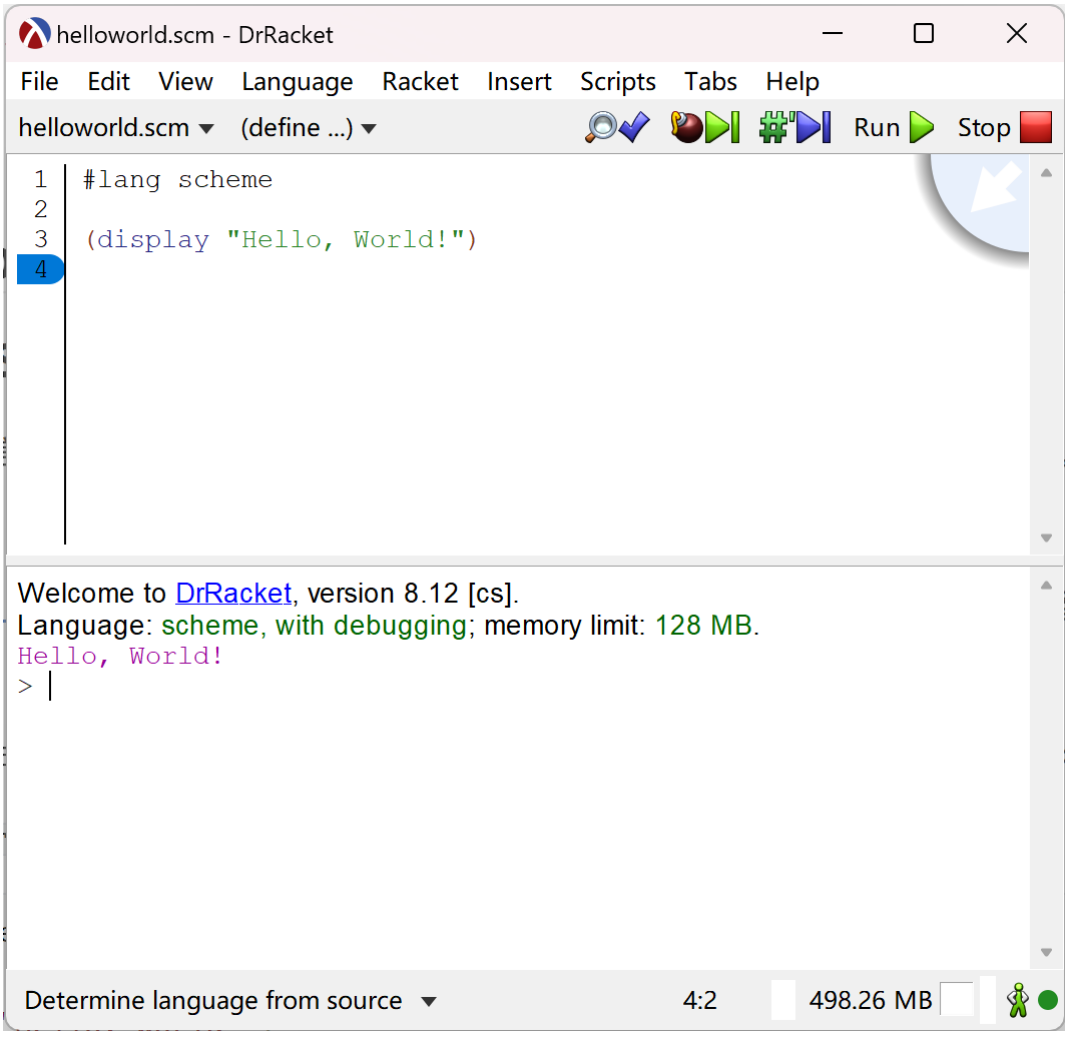
Racket & Scheme

Scheme 编程语言是一种 Lisp 方言，在接下来若干次的作业里我们将会学习并使用这门语言。

有许多免费的 Scheme 解释器实现，本课程使用比较现代的 Racket。

你可以在 Racket 官网 下载到 Racket，助教使用的版本为 Version 8.12 (February 2024) 供参考。

安装成功后，你可以通过 Racket 的图形化集成开发环境 DrRacket 打开源文件编辑或解释执行程序，比如



也可以直接命令行解释执行程序，例如

```
D:\Racket
λ racket C:\Users\Setsuna\Desktop\SICP-FDU\HW0\helloworld.scm
Hello, World!
D:\Racket
```

当然你也可以通过 VSCode 等编辑器来编辑代码。

Hello, World!

HW0 提供了一个基础的 HelloWorld 程序 `HW0/helloworld.scm`：

```
#lang scheme

(display "Hello, World!")
```

其中第一行 `#lang scheme` 表示该程序是 scheme 语言写的（因为 Racket 支持其他语言比如 racket 它自己），如果你使用的解释器不是 Racket（比如 MIT-Scheme）那就不需要这行。

你的任务

你可以自由添加辅助函数，但不要改变待实现函数的签名。

每组文件自带测试样例，可以直接运行来检验自己的实现是否正确。

func.scm

实现以下简单多项式，其中 x 是 -10 到 10 之间的整数。

$$f(x) = 2x^2 + x - 4$$

```
(define (f x)
  'your-code-here)
```

primes.scm

求 $n(2 \leq n \leq 10^7)$ 以内质数的个数，方法自选但复杂度不要超过 $O(n\sqrt{n})$。

取模运算可以使用 `remainder` 函数，如 `(remainder x y)` 。

```
(define (count-primes-in n)
  'your-code-here)
```

exp.scm

实现快速幂取模用于求 $x^y \bmod p (1 \leq x, p \leq 10^9, 0 \leq y \leq 10^9)$，你需要保证实现是尾递归的。

尾递归实现可以参考 Lecture 2 Slides 的第 68 页，其中 Iterative 的版本就是尾递归的，即栈空间是 $O(1)$ 的，而 67 页 Recursive 的版本不是尾递归的，需要 $O(n)$ 的栈空间。

快速幂算法可以参考 Wikipedia。

```
(define (fast-exp-mod x y p)
  'your-code-here)
```

参考资料

1. D. Friedman, M. Felleisen, *The little Schemer (4th ed.)*. MIT Press, 1996. [pdf]
2. H. Abelson, G. Sussman, *Structure and Interpretation of Computer Programs*. MIT Press, 1996. [pdf]