Racket Basics

Installing Dr. Racket

- Windows : https://download.racket-lang.org/
- Debian/Ubuntu : apt install racket
- Other *nix : Download From webpage^

Racket is Functional Programming Language

What is a Functional Programming Language

- Based on lambda calculus
- No concept of state
  - No concept of variables like imperative programming languages
  - Variables can be declared but not reassigned
- A Functional Program is nothing but a function applied on a set of values
  - (func a b c d e …….)
- But a function can be applied on the output of another function
  - (funcA a b (funcB c d)) -> What is the imperative equivalent?

But how are functional programming languages useful?

Are functional programming languages as expressive as imperative programming languages?

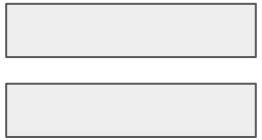
The Church-Turing Thesis

Turing Machine = Lambda Calculus

*Equality here refers to the set of problems that a language can compute

And hence

Imperative Languages



Functional Languages

*Equality here refers to the set of problems that a language can compute

But Practicalities

- In the real world, there is some need of state
  - Can’t live without databases
  - Can’t live without network sockets
- So in real life, ‘impure’ versions of Functional languages are used

Lets start evaluating some expressions

- On your terminal, start racket
- We work with IDLE prompt

(+ 1 2)

(- 1 2)

(+ 1 2 3)

```
(* (+ 1 2) (+ 4 5))
```

Define - Global Variables

- (define x 3)
- (set! x (* x x))

Lists

- ‘(1 2 3 4 5)
- (car ‘(1 2 3 4 5))
- (cdr ‘(1 2 3 4 5))

If else

Syntax : (if test-expr then-expr else-expr)

(if (equal? (remainder 5 2) 1) (* 3 3) (* 2 2))

For loop

For loop only allows iterating through sequence of values (lists, maps)

```
(for ([i ‘(1 2 3 4 5)])

(println i)

)
```

While Loop

While loop syntax :

(while <test> <body>)

You need to include the while loop package

raco pkg install while-loop

A quick challenge

How will you exit this program?

Defining Functions

(define (func-name param1 param2 ... paramN) expr)

(define (square num) (* num num))

Using the function :

(square 3)

Function to calculate factorial of a +ve integer

Sum of all numbers in a list