CPSC 213 Lab 8

Return of 110 MT2

Randy Zhu

Reading function pointers

How do we read function pointers?

- `type1 (f*) (type2 type3)` is a function pointer of a function that maps `type2, type3` to `type1`
  - function definition: `type1 func(type2 x, type3 y) { ... }`
- Concrete example: `void* (f*) (int* int char)` is a function pointer to a function that:
  - Takes an int pointer, integer, and character, and produces a void pointer (pointer to anything)
  - `void* func(int* x, int y, char c) { ... }`

What is m, r, and fi?

Revisiting an old friend

- **m** -> Racket's **m**ap, also typically called **m**ap in Java and other languages
  - Doesn't return anything (void), mutates its parameter *a*
- **r** -> Racket's foldl, called "**r**educe" in most other languages
  - Returns, the value of the accumulator after the foldl
- **fi** -> Racket's **f**ilter, called "filter" in most other languages
  - Returns the number of items in *a* that satisfy *p*

Lab Code

For your credit

ALXN

Q1

Sum two arrays

```
int add(int x, int y) {

    return x + y;

}

int *c = malloc(sizeof(int) * n);

m(add, a, b, c, n); // (map add a b)
```

Q2

Find max of an array

```
int max(int a, int b) {

    return a > b ? a : b;

}

int mx = r(max, c, n, 0); // (foldl max 0 c)
```

Q3

Find and count the number of even/odd elements

```
int parity(int n, int p) {
    int n_is_even = (n % 2 == 0);
    if (p) return !n_is_even;
    else return n_is_even;
}
n_even = fi(parity, a, e, n, 0);
n_odd = fi(parity, b, o, n, 1);
```

Q4

Find minimum

```
int q4(struct q4_args_s args) {

    int min_e = r(min, e, n_even, 2147483647); // (foldl min e)

    int min_o = r(min, o, n_odd, 2147483647); // (foldl min o)

    int mn = min(min_e, min_o); // (min (foldl min e) (foldl min o))
```

Q5

Find all c_i of c such that $mn < c\_i < mx$

```
int global_min, global_max; // these two are GLOBAL scope

int between(int n, int _ignored) {

    if (global_min < n && n < global_max) {

        return 1;

    }

    return 0;

}
```

Q5

Find all c_i of c such that $mn < c\_i < mx$

```
int q5(struct q5_args_s args) {

    global_min = mn; // "hoist" mn and mx up

    global_max = mx;

    int* _ = malloc(sizeof(int) * n);

    int count = fi(between, c, _, n, 0);

    free(_);
```

Racket > C

Isn't this ugly?

- Q1 in Racket: `(map add a b)`
- Q2 in Racket: `(foldl max 0 c)`
- Q3 in Racket: `(length (filter even a))`, `(length (filter odd b))`
- Q4 in Racket: `(min (foldl min e) (foldl min o))`
- Q5 in Racket: `(filter (lambda (n) (and (< mn n) (< n max))) c)`
- 7 slides of code in 5 lines

Want to find out how to make pretty programs?

CPSC 311/411

- You'll learn about how `lambda`, and all your favourite language features are implemented
- Learn how programmers freed (no pun intended) themselves from writing C and assembly