CSE216
Programming Abstraction

Instructor: Zhoulai Fu

State University of New York, Korea

Some old questions

Lambda calculus

Small-1

- (λ x.x) a
- (λ x.y) a
- (λ x.xy) a
- (λ x. yx) a
- (λ x. xx) a
- (λ x. yy) a

Small-2

- (λ x.x) a b
- (λ x.y) a b
- (λ x.xy) a b
- (λ x. yx) a b
- (λ x. xx) a b
- (λ x. yy) a b

Small-3

- (λ x.x) λ a. b
- (λ x.y) λ a. b
- (λ x.xy) λ a. b
- (λ x. yx) λ a. b
- (λ x. xx) λ a. b
- (λ x. yy) λ a. b

Small-4

- (λ x.x) x
- (λ x.y) x
- (λ x.xy) x
- (λ x. yx) x
- (λ x. xx) x
- (λ x. yy) x

Small-5

- (λ x.x) x y
- (λ x.y) x y
- (λ x.xy) x y
- (λ x. yx) x y
- (λ x. xx) x y
- (λ x. yy) x y

Exercise: beta reduction

- (λz.z) (λz.z z) (λz.z q)

Exercise: beta reduction

- (λs.λq.s q q) (λa.a) b

Exercise: beta reduction

- (λs.λq.s q q) (λq.q) q

Exercise: beta reduction

- ((λs.s s) (λq.q)) (λq.q)

Exercise: beta reduction

- (λ x.λy.x) x y

Exercise: beta reduction

- (λ x. λ y. λ z. y (w y x)) λ s. λ z. z

Exercises:
Context-Free Grammar

Exercise 3

Given the grammar G with the following productions:

- $S \to aSb$
- $S \to \epsilon$

Determine the language $L(G)$ generated by G.

Exercise 4

Given the grammar G with the productions:

- $S \to aSa$
- $S \to bSb$
- $S \to \epsilon$

What is the language $L(G)$?

Exercise 5

1

Consider the grammar G defined as:

- $S \to aS$
- $S \to Sb$
- $S \to \epsilon$

Define the language $L(G)$.

2. Find a word that does not belong to L(G)

Exercise 6

Create a grammar that generates the language of all strings of the form:

- a language containing only the words "dog", "cat", and “fish”.

Exercise 7

Create a grammar that generates the language of all strings of the form:

- "a^n", where n ≥ 0.

Exercise 8

Create a grammar that generates the language of all strings of the form:

- "a^n b^m", where n, m ≥ 0.

Exercise 9

Create a grammar that generates the language of all strings of the form:

- "a^n b^n", where n ≥ 0.

Exercise 10

Create a grammar that generates the language of all strings of the form:

- all strings over {a, b} that start with 'a' and end with 'b'.

Ocaml

- Make sure you have no problem solving **#1-#10** in 99 problems: https://v2.ocaml.org/learn/tutorials/99problems.html

A harder one

Some programming languages (like Python) allow us to quickly *slice* a list based on two integers `i` and `j`, to return the sublist from index `i` (inclusive) and `j` (not inclusive). We want such a slicing function in OCaml as well.

Write a function **`slice`** as follows: given a list and two indices, `i` and `j`, extract the slice of the list containing the elements from the $\texttt{i}^{\text{th}}$ (inclusive) to the $\texttt{j}^{\text{th}}$ (not inclusive) positions in the original list.

```
# slice ["a";"b";"c";"d";"e";"f";"g";"h"] 2 6;;
- : string list = ["c"; "d"; "e"; "f"]
```

Invalid index arguments should be handled *gracefully*. For example,

```
# slice ["a";"b";"c";"d";"e";"f";"g";"h"] 3 2;;
- : string list = []
# slice ["a";"b";"c";"d";"e";"f";"g";"h"] 3 20;
- : string list = ["d";"e";"f";"g";"h"];
```

You do *not*, however, need to worry about handling negative indices.