

# W2 - IPS

Nikolaj Gade (qhp695)

May 2022

## Task 1

1)

a)

The string must consist of any number of pairs of 'o's and 'g's. The pairs can be any combination of 2 characters, both of which are either 'o' or 'g'. Thus, each character can be written as, 'o|g', and each pair as '(o|g)(o|g)'. The full string can be written as:

$$((o|g)(o|g))^*$$

c)

Keeping with the idea that an even number of characters can be split up in "pairs" of characters, an odd number of 'o's or 'g's *must* result in an odd number of both. Thus, the string will either consist of an even number of 'o's, followed by an even number of 'g's, **or** an even number of 'o's, followed by 'og', followed by an even number of 'g's. The full string can be written as:

b)

$$(oo)^*(og)?(gg)^*$$

The first character must be 'o', and it must be followed by a character that is either 'o' or 'g'. Following that, the answer is the same as the previous problem:

$$o(o|g)((o|g)(o|g))^*$$

2)

a)

are placed 2 at a time.

'a's and 'b's are placed at the same time, using the  $T$  starting symbol. After all 'a's and 'b's are placed, the symbol turns to 'S', which places at least 1 'c'.

$$T = \begin{cases} aTbb \\ abb \end{cases}$$

$$T = \begin{cases} aTb \\ S \end{cases}$$

$$S = \begin{cases} c \\ cS \end{cases}$$

c)

Once again, 'a's and 'b's are placed at the same time with the starting symbol  $T$ . Afterwards, any amount of 'a's are placed before the 'b's.

$$T = \begin{cases} aTb \\ S \end{cases}$$

$$S = \begin{cases} aS \end{cases}$$

b)

Like with the previous problem, both the 'a's and the 'b's are placed at the same time, but the 'b's

3)

a)

`%nonassoc letprec` designates the `let` token as being non-associative, which means ambiguous implementations will lead to a syntax error.

b)

The order of the associativity declarations provide the precedence for the operators. So in the current way the code is written, the line `let x = 10 in x + 10 > 15` will be parsed as `let x = 10 in (x < 15)`, but if , it would be parsed as `(let x = 10 in x) < 15`.

c)

The code `{ Let (Dec (fst $2, $4, $3), $6, $1) }` creates a `Let` instance, which contains the declared variable, the following expression, as well as the keyword.

## Task 2

See code.

## Task 3

a)

```
filter (('a -> bool) * ['a]) -> ['a]
```

b)

```
1 CheckExp(exp, vtable, ftable) = case exp of
2   filter(p, arr_exp) =>
3     let array_type = CheckExp(arr_exp, vtable, ftable)
4     let element_type = match array_type with
5       | Array(type) -> type
6       | _ -> Error()
7
8     let function_type = lookup(ftable, name(p))
9     match function_type with
10    | unbound -> Error()
11    | (input_type, output_type) ->
12      if input_type == element_type && output_type == bool then
13        Array(element_type)
14      else Error()
15    | _ -> Error()
```

## Task 4