Soyeon Kim, 106292803

Homework 4

CS 131, Fall 2024
Carey Nachenberg

PYTH9

```python
def strip_characters(sentence, chars_to_remove):
   return "".join([s for s in sentence if s not in chars_to_remove])
```

PYTH10

Python supports closures. A closure is created when a nested function captures the variables from its enclosing scope.

```python
def foo(a):
   def bar(b):
       return a + b # captures a from the outer scope
   return bar
print(foo("1")("2")) # print 12
```

PYTH11

Part A

```python
def convert_to_decimal(bits):
   exponents = range(len(bits)-1, -1, -1)
   nums = [b * 2**e for b, e in zip(bits, exponents)]
   return reduce(lambda acc, num: acc + num, nums)
```

Part B

```python
def parse_csv(lines):
   return [(word, int(num)) for word, num in (line.split(",") for line in lines)]
```

Part C

```python
def unique_characters(sentence):
   return {c for c in sentence}
```

Part D

```python
def squares_dict(lower_bound, upper_bound):
   return {i: i**2 for i in range(lower_bound, upper_bound+1)}
```

Soyeon Kim, 106292803

HASK17

Part A

```
int longestRun(std::vector<bool> arr) {
 int max = 0;
 int count = 0;

 for (bool a : arr) {
   if (a) {
     count++;
   }
   max = std::max(max, count);
   if (!a) {
     count = 0;
   }
 }
 return max;
}
```

Part B

```
longest_run :: [Bool] -> Int
longest_run ls = longest_run_helper ls 0 0
 where
   longest_run_helper [] _ m = m
   longest_run_helper (x:xs) curr m
     | x = longest_run_helper xs (curr + 1) (max (curr + 1) m)
     | otherwise = longest_run_helper xs 0 (max curr m)
```

ChatGPT's answer:

```
longest_run ls = fst (foldl (\(m, curr) x -> if x then ((max (curr+1) m), curr+1) else
(m, 0)) (0, 0) ls)
```

Part C

```
#include <queue>
unsigned maxTreeValue(Tree *root) {
 if (root == nullptr) return 0;

 unsigned max = root->value;
 queue<Tree *> queue;
 queue.push(root);

 while (!queue.empty()) {
   Tree* curr = queue.front();
   queue.pop();
   max = std::max(max, curr->value);
```

```
    for (Tree *child : curr->children) {
      queue.push(child);
    }
  }
  return max;
}
```

Part D

```
max_tree_value :: Tree -> Integer
max_tree_value Empty = 0
max_tree_value (Node v []) = v
max_tree_value (Node v (x:xs)) = max v (max (max_tree_value x) (max_tree_value (Node v
xs)))
```

ChatGPT's answer:

```
max_tree_value (Node value child) = max value (maximum (map max_tree_value child))
```

HASK18

```
super_giuseppe :: [Event] -> Integer
super_giuseppe events = super_giuseppe_helper events 100 False
 where
   super_giuseppe_helper [] hp _ = hp
   super_giuseppe_helper (event:rest) hp mode
     | hp <= 0 = -1
     | otherwise =
         let (new_hp, new_mode) = handle_event event hp mode
         in super_giuseppe_helper rest new_hp new_mode
   handle_event (Travel n) hp True = (hp, True)
   handle_event (Travel n) hp False = (min 100 (hp + (n `div` 4)), False)
   handle_event (Fight n) hp True = (hp - (n `div` 2), hp - (n `div` 2) <= 40)
   handle_event (Fight n) hp False = (hp - n, hp - n <= 40)
   handle_event (Heal n) hp _ = (min 100 (hp + n), min 100 (hp + n) <= 40)
```

HASK19

Part A

```
sumSquares 0 = 0
sumSquares n = n^2 + sumSquares (n-1)
```

Part B

```
sumSquares 0 = 0
sumSquares n = sumSquares_helper n 0
 where
```

```
sumSquares_helper 0 acc = acc
sumSquares_helper n acc = sumSquares_helper (n-1) (acc + n^2)
```

DATA1

Part A

It is a dynamically typed language because `user_id` was set to string type and then changed to int type. The variable type of statically typed language is fixed and cannot change.

Part B

1. It is a dynamically typed language because `y` was set to integer 10 and then changed to double 3.5, but it did not get a compile error.
2. Conversion is performed in lines 2, 4, and 9, and it’s narrowing because int and double are converted to a string that is unrelated.
3. It will throw a compile error.
4. It is a dynamically typed language because `y` has changed from integer to string type.

DATA2

Part A

It says that C++ is a weakly typed language. `w.f` is not initialized but the program allows access to it.

Part B

Zig is a strongly typed language, where only the active field in a union can be accessed or modified. On the other hand, C++ allows access to any member.
C++ is good for low-level memory manipulation but it is very vulnerable to memory management and causes errors. I think strongly typed language is better for most high-level applications.

DATA3

Part A

- Conversion
  - line 6: long → int, narrowing
  - line 17: short → long - widening, type promotion
  - line 18: short → double, narrowing
- Cast
  - line 11: Student → Person - upcast
  - line 19: Person → Student - downcast

Part B

- Conversion
  - line 9: int → int

    - line 10: int → float: narrowing
    - line 21: int → float: narrowing
- Cast
    - line 18: Student → Person