CMSC330 Spring 2025 Quiz 2

Name: _____ **UID:** _____

Section Number: _____ **Proctoring TA:** _____

Problem 1: Basics

[Total 4 pts]

	True	False
∃ a regular expression that describes strings of any length that contain any number of balanced parentheses	(T)	(F)
Property Based Testing is intended to completely replace unit testing	(T)	(F)
The expression `(fun x y -> 3) (print_string "a") (print_string "b")` always prints out "ab"	(T)	(F)
Regular expressions can only describe a finite set of strings	(T)	(F)

Problem 2: Property Based Testing

[Total 6 pts]

Consider the following incorrect `tree_map` function.

```
type tree = Leaf of int | Node of int * tree * tree

(* has bug(s)! *)
let rec tree_map f tree = match tree with
    Leaf(x) -> Leaf(x)
    |Node(x,l,r) -> Node(f x, l, r)
```

Consider the following property *p* about the `tree_map` function:

p : calling `tree_map` using the identity function should not change the tree

Using a correct implementation of `tree_map`, this property *p* should hold true for all valid inputs?

(Yes) (No)

Using our implementation of `tree_map`, this property *p* would **not** hold true for all valid inputs?

(Yes) (No)

Suppose I encode this property in OCaml to be used in OCaml's QCheck library as the following:

```
let prop f tree = tree_map (fun x -> x) tree = tree
```

The above `prop` function is a valid encoding of the property *p*. (Yes) (No)

Problem 3: Regex

[Total 10 pts]

*	zero or more repetitions of the preceding character or group
+	one or more repetitions of the preceding character or group
?	zero or one repetitions of the preceding character or group
.	any character
$r_1|r_2$	r_1 or r_2 (eg. a|b means 'a' or 'b')
[abc]	match any character in abc
[^r_1]	anything except r_1 (eg. [^abc] is anything but an 'a', 'b', or 'c')
[r_1-r_2]	range specification (eg. [a-z] means any letter in the ASCII range of a-z)
{n}	exactly n repetitions of the preceding character or group
{n,}	at least n repetitions of the preceding character or group
{m,n}	at least m and at most n repetitions of the preceding character or group
^	start of string
$	end of string

(a) Names and Ages

[4 pts]

Write a regex that describes a person's name and their age in the format:

```
Name: age
```

- A Name starts with two Capital Letters followed by any number of lowercase letters
- An age is a valid integer from 50 - 92(inclusive) (can be 0 padded)

Valid Examples of Names and Ages names

```
BAka: 84
MAmat: 92
WAluiji: 0050
```

Invalid examples of Names and Ages names

```
nocaptial: -34
Hasnumber6: 200
NOinteger: 63.14
```

(b) Addresses

[6 pts]

Write a regex that describes **exactly** a youtube video URL.

- Will always start with: `youtube.com/watch?`
- Will be followed with 1 or more key-value pairs in the form: `key=value`
- Keys will be single lowercase letters
- Values will be any alphanumeric (lowercase, uppercase, digits), each at least 1 character long
- key-value pairs will be separated by an ampersand (&) character if there is more than 1 pair

valid urls

```
youtube.com/watch?v=dQw4w9WgXcQ
youtube.com/watch?v=XqZsoesa55w&t=10
youtube.com/watch?a=0&v=k85mRPqvMbE
```

invalid urls

```
youtube.com/watch?autoplay=1&video=djymZspawFc
youtube.com/watch?
youtube.com/watch?t=20v=Zq1QJ2QztgM
```

`youtube\.com/watch\?`