Protokoll Klausur WS 19/20 Informatik I – Osterman

P R O G R A M M I E R F R A G E N

1a. In einer Firma gibt es departments und sub-departments mit einer Anzahl an Mitarbeiter. In der Unterabteilung kann wieder eine Unterabteilunf existieren usw. Man muss eine Datendefinitionbestimmen und Beispielwert.

1b. sum-employees auch mithilfe von foldr oder map

2a. Man hat einen Baum vom Typ X und man hat entweder ein branch oder ein leaf mit dem Wert

(define-struct branch (left middle right))

(define-struct leaf (label))

Man muss fold-tree implementieren mit (define (fold-tree f l t) …) wobei f auf die branches und l auf die leaves angewendet werden sollen.

2b. fold-tree aus a verwenden um Funktion zu implementieren, die alle Werte zusammenmultipliziert (mit Signatur!)

2c.fold-tree aus a verwenden um Funktion zu implementieren, die eine Liste zurückgibt, wo alle Werte quadriert vorliegen (mit Signatur!)

T H E O R I E F R A G E N

(define-struct collision (oben unten rechts links)

; A collision is a struct: (make-collision boolean boolean boolean boolean)

1. Was beschreibt die erste Zeile des Programms?

a. Produkttyp mit Namen collision

b. Strukturtyp mit Namen collision

c. Strukturdefinition mit Namen collision

d. keine der Antworten

2. Was ist eine collision?

a. rekursiver Datentyp

b. Summentyp

c. Intervalltyp

d. Produkttyp

3. Was ist eine gültige Collision?

a.<make-collision #t 42 #t „hello“>

b.<make-collision #t #f #f #t #t>

c. <make-collision #t #t #t #t>

d. <make-collision #t <make-collision 42 #t #f #t> 42 „hello“ #t>

3b. Wie viele Werte kann collision bekommen?

a.4

b.1

c.16

d.unendlich viele

4. Wann kommen die ersten Tests gemäß des Entwurfsrezepts?

a. Nach den Templates

b. Nach der Implementierung

c. Nach den Datenbeispielen

d. Nach dem Funktionskopf

5. Was ergibt (#true + 3)?

a. Fehler: erwartet Zahl als erstes Argument

b. Wert #t

c. Fehler: erwartet Funktion nach öffnender Klammer

d. …

Folgende kontextfreie Grammatik:

<TLD> ::= de | com | co.uk

<PREFIX> ::= www | https://www | hhtp://www

<NAME> ::= google | amazon| webshop | home | <Name>.<Name>

<URL> ::= <Prefix> . <Name> . <TLD> /

6. Welche ist nicht gültig?

a. https://www.amazon.google.com/

b.http://www.google.com/

c. https://www.amazon.com/home/webshop/

d. …

Folgendes Programm:

```
(define (myfun x)
    (cond [(false? X)   x]
          [else         (+ 1 x)]))
```

7. Signatur?

a. Boolean -> Number

b. MaybeNumber -> MaybeNumber

c. MaybeNumber -> Number

(d. Number -> Boolean) ?

8. Auswertung von (myfun (myfun 2))?

a. #f

b. 2

c. 4

d. Fehler

unfold hat Signatur:

; [X Y] (Y -> Boolean) (Y -> Y) (Y -> X) Y -> (List Of X)

9. drittes Argument ist eine Funktion, die…

a.

b.

c.

d.

10. Signatur von compose?

a.

b.

c.

d.

11. Welche Konstantendefinition wird bei der Auswertung von (define f (lambda (x) (+ 1 2))) der Umgebung hinzugefügt?

a. (define (f x) 3)

b. (define f (lambda (x) 3))

c. (define f 3)

d. (define f (lambda (x) (+ 1 2)))

Folgendes Programm:

(define (forbidden-word? Sentence word) …)

(define (any-forbidden-words? Sentence words)

(and (cons? Words)

(or (forbidden-word? Sentence (first words))

…)))

12. Lücke muss mit welchem Ausdruck gefüllt werden?

a. (any-forbidden-words? Sentence (rest words))

b. (any-forbidden-words? (rest sentence) words)

c. (forbidden-word? (rest sentence) words)

d. (forbidden-word? Sentence (rest words))

Folgendes Programm:

```
(local [(define x 1)]
(local [(define x 2)]
3))
(+ x x)
```

13. Auswertung?

a. Fehler

b. 2

c.

d.

14. Was ist syntaktischer Zucker?

a. Konzepte einer Sprache, die für dir Programmausführung unwesentlich sind

b. Konzepte einer Sprache, die durch andere Konzepte ausgedrückt werden könne

c. Konzepte einer Sprache, die nicht direkt einem Prozessorbefehl entsprechen

d. Konzepte einer Sprache, die von einem Editor zusätzlich zum Programm angezeigt werden müssen

Folgendes Programm:

```
(define (my-animate f)
  (big-bang 0
            [to-draw f]
            [on-tick …]))
```

15. Signatur?

a. Number -> Number

b. (Image -> Number) -> Image

c. Image -> Number

d. (Number -> Image) -> Number

Folgendes Programm:

```
(define (f b1 b2)
(and b1 (or (not b2) (not #t))))
```

16. Vereinfachung durch Refactoring führt zu:

a. (not (and b1 b2))

b. (and b1 (not b2))

c.(or b1 b2)

d.(and b1 b2)

17. Was ist eine Umgebung?

a.

b.

c.

d.

18. Funktion erster Ordnung?

a. (lambda (x) (lambda (y) y))

b. (lambda (x) (lambda (y) (x y)))

c. (lambda (x) ((lambda (y) y) x))

d.

19. Nächster Schritt bei Auswertung von:

```
(cond [#false (+ 2 2)]
      [(> 1 0) (+ 1 1)])
```

a. (cond [(> 1 0) (+ 1 1)])

b. 2

c. (+ 1 1)

d.

20. Mehrere Werte zu einem zusammenzufassen?

a. Konstanten

b. Strukturen

c. Produkttypen

d. Summentypen

21. Was ist eine Umgebung für (define (f x) (+ x 1))?

a. (define y 6)

b. (define y (+ x 1))

c. (f 5)

d. (define y x)

22. DRY Prinzip, besagt, dass….

a.

b.

c.

d.

Folgendes Programm:

(not (and (> 3 5) (+ 5 2)))

23. Auswertung, ergibt…?

a.

b.

c.

d.

24. Was sind magic numbers?

a.

b.

c.

d.

25. Was bedeutet `(+ ,(+ 1 2) ,(+ 3 4))?

a. (list ´+ 3 7)

b.

c.

d.

26. Was passiert beim refactoring?

27. gültiger Wert?

a. 8

b. (define x 2)

c. (define (f x) …)

d. (string-append „hello“ „world“)