

Пример экзаменационной письменной работы прошлых лет

Тесты

Указание: для ответа на вопросы сделайте отметки в ☐ рядом с правильными вариантами ответов (верными могут быть один или более вариантов).

I. Дано определение: `(define (x y z) (z y x))`

Укажите результат вычисления выражения: `(x (and) (lambda (a b) (or a b)))`

a) ☐ ошибка b) ☐ #t c) ☐ #f d) ☐ #<procedure:x>

II. Укажите только верные утверждения относительно выражений в Scheme:

- a) ☐ литералы являются одним из видов выражений;
- b) ☐ литеры являются одним из видов литералов;
- c) ☐ литералы являются одним из видов литер;
- d) ☐ литеры являются одним из видов выражений.

III. Укажите только верные утверждения относительно связываний, кадров и окружений:

- a) ☐ окружение содержит не более одного связывания для любого имени;
- b) ☐ кадр содержит не более одного связывания для любого имени;
- c) ☐ окружение содержит произвольное количество связываний для любого имени;
- d) ☐ количество связываний для имени в окружении не превышает количество кадров, входящих в окружение.

IV. Пусть есть определения: `(define a (list * 1 2 3))`

`(define b (append (list * 1 2) (cdddr a)))`

Укажите выражения, являющиеся истинными:

a) ☐ `(= a b)` b) ☐ `(eq? a b)` c) ☐ `(equal? a b)` d) ☐ `(= (caddr a) (caddr b))`

V. Укажите только верные утверждения о макросах:

- a) ☐ при использовании макроса сначала вычисляются все его аргументы;
- b) ☐ при использовании макроса сначала вычисляются некоторые его аргументы;
- c) ☐ при использовании макроса никакие его аргументы не вычисляются;
- d) ☐ макросом можно описать собственную спецформу;
- e) ☐ макросом нельзя описать собственную спецформу.

VI. Укажите наиболее полный список того, что может быть явно представлено на диаграммах классов:

- a) ☐ классы и связи между классами;
- b) ☐ классы, атрибуты, операции и связи между классами;
- c) ☐ классы, атрибуты, операции, экземпляры классов и связи между классами;
- d) ☐ классы, атрибуты, операции, экземпляры классов, связи между классами и связи между экземплярами классов.

VII. Укажите верные утверждения о λ -выражениях:

- a) ☐ у любого λ -выражения существует нормальная форма;
- b) ☐ у λ -выражения может не быть нормальной формы;
- c) ☐ нормальная форма любого λ -выражения единственна;
- d) ☐ к λ -выражению в нормальной форме никакая β -редукция не может быть применена.

VIII. Укажите верные утверждения о модели вычислений с окружениями:

- a) ☐ порядок вычисления комбинации полностью не определён;
- b) ☐ порядок вычисления комбинации полностью определён;
- c) ☐ порядок вычислений не имеет значения;
- d) ☐ порядок вычислений имеет значение.

IX. Отметьте все пункты, в которых указаны спецформы:

a) ☐ `append`; c) ☐ `if`; e) ☐ `begin`; g) ☐ `apply`;
b) ☐ `list`; d) ☐ `let`; f) ☐ `quote`; h) ☐ `lambda`.

X. Укажите все верные утверждения о спецформе `cond`:

- a) ☐ результат вычисления `cond` всегда определён;
- b) ☐ результат вычисления `cond` может быть не определён;
- c) ☐ результат вычисления `cond` с `else`-альтернативой всегда определён;
- d) ☐ в `cond` обязательно должны быть две или более альтернативы;
- e) ☐ в `cond` обязательно должна быть хотя бы одна альтернатива;
- f) ☐ в `cond` обязательно должна быть одна `else`-альтернатива.

Задачи

Указание: В решениях задач можно применять присваивание там, где это необходимо. Эффективность решения задач по времени счёта и памяти учитывается. Правильное, но неэффективное решение может быть оценено ниже максимальных баллов. Решение с присваиванием, без которого можно обойтись, также может быть оценено ниже максимума. При использовании в решении дополнительных функций обязательны комментарии, поясняющие их назначение и работу.

2.I. Опишите макрос (`when test expr1 ... exprN`), работающий как «сокращённый `if`». То есть, сначала вычисляется проверка `test`. Если значение `test` истинно, то вычисляются `expr1 ... exprN`, и значением `when` является значение `exprN`, или `#t` при отсутствии `expr1 ... exprN`. Если значение `test` ложно, то выражения `expr1 ... exprN` не вычисляются, и значением `when` является `#f`. Описывая макрос, используйте `if` и `begin`.

2.II. Дополните предложенные описания функции, переворачивающей список. `foldl` и `foldr` – лево- и право- ассоциативные свёртки. Дополняя описания, можно пользоваться другими функциями работы со списками. Нельзя определять вспомогательные функции и использовать внутри дополнения `foldr`, `foldl`, `map` и т. п.

```
(define (reverse1 lst)
```

```
  (foldl (lambda (x y) _____
```

```
    _____
```

```
  '() lst))
```

```
(define (reverse2 lst)
```

```
  (foldr (lambda (x y) _____
```

```
    _____
```

```
  '() lst))
```

2.III. Реализуйте функцию (`dec2bin n`), которая возвращает число, десятичная запись которого совпадает с двоичной записью числа `n`. Реализация должна порождать итеративный процесс. Пример: (`dec2bin 7`) => 111

2.IV. Опишите функцию (`different lst`), возвращающую количество различных элементов в списке чисел `lst`. Пример: (`different (list 1 2 1 3 1 4)`) => 4

2.V. Найдите нормальную форму $(\lambda x. (\lambda y. (+ x ((\lambda x. (- x 3)) y))))$ 5 6

Обоснуйте ответ последовательным применением редукций. Примечание: `+`, `-` и числа имеют обычный смысл. Если у этого λ -выражения нет нормальной формы, обоснуйте, почему это так.

2.VI. Пусть вычислено:

```
(define a (list 1 (list 2 (list 3))))
```

```
(define b (list 4 'a a))
```

```
(define c (cdr a))
```

```
(set-cdr! c (cdr b))
```

нарисуйте стрелочную диаграмму, изображающую `a`, `b` и `c` после выполненных вычислений.

2.VII Опишите функцию (`perm n`), возвращающую список из всех различных перестановок натуральных чисел от 1 до `n`. Каждая перестановка в списке является списком. Пример (порядок перестановок в списке может быть иным):

```
(perm n) => '((1 2 3) (1 3 2) (2 1 3) (2 3 1) (3 1 2) (3 2 1))
```

Фамилия И. О.: _____ Группа: _____ дд.мм

Текстовые вопросы:

Указание: В ответах на текстовые вопросы засчитываются только Ваши собственные осмысленные оригинальные примеры, которые существенно отличаются от примеров, рассмотренных в лекциях и/или в учебнике.

3.I. (<=4 балла) Расскажите об остаточных вычислениях. Приведите пример остаточных вычислений (с кодом). Расскажите об описании функций в стиле передачи остаточных вычислений (cps continuation-passing style). Приведите пример cps-описания нетривиальной функции. Какие возможности предоставляют cps-описания функций по сравнению с обычными?

3.II. (<= 3 балла) Что такое хвостовая рекурсия? Приведите код нетривиальной функции, в котором используется хвостовая рекурсия, а также код той же функции без использования хвостовой рекурсии. В чём преимущество использования хвостовой рекурсии?

3.III. (<= 3 балла) Расскажите о векторах в Scheme (vector). Расскажите о функциях для работы с векторами (make-vector, vector-ref, vector-set!). В чём преимущество использования векторов вместо списков? Проиллюстрируйте это преимущество осмысленным примером кода.