Тестовое задание для диагностического тестирования по дисциплине:

Функциональное программирование

Квалификация выпускника	бакалавр *бакалавр, магистр, специалист*
Направление подготовки	09.03.04 *шифр* Программная инженерия *наименование*
Направленность (профиль)	Программное обеспечение компьютерных систем *наименование*
Форма обучения	очная *наименование*
Кафедра-разработчик	Автоматики и компьютерных систем *наименование*
Выпускающая кафедра	Автоматики и компьютерных систем *наименование*

Диагностический тест по дисциплине «Функциональное программирование»

Проверяемые компетенции	Задание	Варианты ответов	Тип сложности	Количество баллов за правильный ответ
ПК-4.2, ОПК-6.1	1) Какая из перечисленных операций составляет основу лямбда-исчисления?	1) импликация 2) абстракция 3) репликация 4) дедукция	легкий	2
ПК-4.2, ОПК-6.1	2) Какая из перечисленных операций составляет основу лямбда-исчисления?	1) индукция 2) аппликация 3) репликация 4) симплификация	легкий	2
ПК-4.2, ОПК-6.1	3) Существует ли нормальная форма для любого лямбда-терма?	1) да, для любого 2) нет, ни для какого 3) существует для небольших лямбда-термов 4) существует, но для некоторых, алгоритм определения которых неизвестен	легкий	2
ПК-4.2, ОПК-6.1	4) Какая стратегия редукции гарантирует приведение к нормальной форме лямбда-терма при ее наличии? Вначале преобразовывать:	1) самый левый из самых внешних редексов 2) самый левый из самых внутренних редексов 3) самый быстрый из всех внешних редексов 4) самый правый из самых внутренних редексов	легкий	2
ПК-4.2, ОПК-6.1	5) Лямбда-исчисление – это исчисление (выберите наиболее полное определение)	1) исчисление греческих букв 2) исчисление в анонимных функциях 3) основа языка программирования 4) формальная система для анализа понятия вычислимости	легкий	2
ПК-4.2, ОПК-6.1	6) Какие из перечисленных операций (структурных элементов) не относятся к функциональному программированию?	1) рекурсия 2) присваивание 3) композиция функций 4) цикл	средний	5
ПК-4.2, ОПК-6.1	7) Какие основные способы борьбы со сложностью используются в	1) функциональная абстракция и функциональная декомпозиция	средний	5

	функциональных программах (выберите один или несколько вариантов)?	2) наследование и полиморфизм 3) функциональная абстракция и мемоизация 4) функциональная декомпозиция и динамическое связывание		
ПК-4.2, ОПК-6.1	8) Какие языки программирования являются существенно поддерживают функциональное программирование (выберите все подходящие варианты)?	1) C++ 2) Java 3) C# 4) Haskell 5) F# 6) FORTH 7) Common Lisp	средний	5
ПК-4.2, ОПК-6.1	9) Какая алгоритмическая модель лежит в основе функционального программирования (выберите один или несколько вариантов)?	1) лямбда-исчисление 2) логика предикатов 1-го порядка 3) логика высших порядков 4) машина Тьюринга	средний	5
ПК-4.2, ОПК-6.1	10) В чем отличия функционального программирования и императивного (выберите один или несколько вариантов)?	1) функциональное программирование оперирует функциями и их применением к данным, императивное – операторами и тем, как они изменяют состояние памяти 2) в функциональном программировании каждая функция может оперировать только с той областью памяти, которая для нее выделена 3) в функциональном программировании происходит автоматический поиск решения задачи по ее декларативному описанию 4) все вышеперечисленное	средний	5
ПК-4.2, ОПК-6.1	11) Какой принцип построения функциональных программ? Выберите наиболее строгое определение.	1) программа – это набор функций, которые преобразует входные данные в выходные, при этом функции также могут рассматриваться как данные 2) программа строится из набора вызывающих друг друга подпрограмм (процедур и функций) 3) программа представляет собой одно большое математическое выражение	средний	5

		4) программа – это набор функций, которые преобразует входные данные в выходные, при этом существует четкое разделение между данными и функциями		
ПК-4.2, ОПК-6.1	12) За счет чего функциональные программы потенциально более надежны? Выберите наиболее строгое определение.	1) на функциональных языках автоматически контролируются ошибки типа переполнения буфера 2) функциональные программы короче 3) функциональные программы содержат минимум побочных эффектов 4) функциональные программ более просты и понятны для программиста	средний	5
ПК-4.2, ОПК-6.1	13) Лексическое замыкание – это	1) невычисленная функция 2) функция, которая также содержит ссылки на лексическое окружение, существовавшего на момент определения функции 3) указатель на функцию 4) совокупность переменных, которые использует функция	средний	5
ПК-4.2, ОПК-6.1	14) Выберите верные соответствия способов передачи аргументов в функции в языках программирования порядкам исчисления	1) вызов по имени – аппликативный порядок исчисления 2) вызов по имени – нормальный порядок исчисления 3) вызов по значению – аппликативный порядок исчисления 4) вызов по значению – нормальный порядок исчисления	средний	5
ПК-4.2, ОПК-6.1	15) Какие из перечисленных языков программирования вобрали в себя парадигму функционального программирования?	1) Python 2) Forth 3) Pascal 4) C 5) Scheme	средний	5
ПК-4.2, ОПК-6.1	16) Выберите все исчислимые формы языка Common Lisp	1) `10` 2) `(2 * 2) + 2`	высокий	8

		3) `(+ (2 * 2) 1)` 4) `(+ 2 2 1)` 5) `(+ ((+ 2 2) 1)`		
ПК-4.2, ОПК-6.1	17) Расставьте формы в порядке увеличения длины списка, который вернет форма, после ее вычисления	1) `(list 1 '(1 2 3 4))` 2) `(cons 1 '(1 2 3 4))` 3) `(member 1 '(1 2 3 4 5 6))` 4) `(append '(1 2) '(34567))`	высокий	8

ПК-4.2, ОПК-6.1	18) Какие значения вернет определенная ниже функция при передаче ей следующих аргументов:
	`(defun f(x)`
	`(if (null x)`
	`nil`
	`(cons (f (cdr x))`
	`(cons (car x) nil))))`
	
	`(f '(a b c))`
	`(f '(nil 2))`
		1) `(C B A)`
		2) `(((NIL C) B) A)`
		3) `(A B C)`
		4) `(NIL 2)`
		5) `((NIL 2) NIL)`
		6) `(2 NIL)`
			высокий	8

ПК-4.2, ОПК-6.1	19) Выберите реализации рекурсивной функции, получающей список (четной длины) чисел и возвращающей список пар исходных элементов.	1)
		`(defun f(ls)`
		`(if (null ls)`
		`ls`
		`(cons (cons (car ls)`
		`(cons (cadr ls)`
		`nil))`
		`(f (cddr ls)))))`
		2)
		`(defun f(ls)`
		`(if ls`
		`(cons (list (car ls) (cdar ls))`
		`(f (cddr ls))))))`
		3)
		`(defun f(ls)`
		`(if (null ls)`
		`ls`
		`(cons (cons (car ls)`
		`(cons (cadr ls)`
		`nil))`
		`(f (cdr ls)))))
			высокий	8

		4)
		`(defun f(ls)`
		`(if (null ls)`
		`ls`
		`(cons (cons (car ls)`
		`(cons (cdar ls)`
		`nil))`
		`(f (cddr ls)))))`
		5)
		`(defun f(ls)`
		`(if ls`
		`(cons (list (car ls) (cadr ls))`
		`(f (cddr ls))))))`		

ПК-4.2, ОПК-6.1	20) Выберите все верные результаты вычислений	1)
		`(reduce #'cons '(nil nil nil))=>((nil))`
		2)
		`(reduce #'append '((1) nil (2)))=>(1 (nil) 2)`
		3)
		`(reduce #'expt '(2 1 2 3))=>12`
		4)
		`(reduce #'* '(2 1 2 3))=>12`
		5)
		`(reduce #'append '((1) nil (2)))=>(1 2)`
		6)
		`(reduce #'expt '(2 1 2 3))=>24`
		7)
		`(reduce #'cons '(nil nil nil))=>(nil)`
		8)
		`(reduce #'* '(2 1 2 3))=>6`
		9)
		`(reduce #'expt '(2 1 2 3))=>64`
			высокий	8

	Итого:			100