Компонент ОПОП 01.03.02 Прикладная математика и информатика. профиль Системное программирование и компьютерные технологии
Б1.О.15.05

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

Дисциплины Функциональное и логическое программирование

Разработчик (и):
Беляев Владимир Яковлевич, доцент кафедры высшей математики и физики
канд. ф.-м. наук, доцент

Утверждено на заседании кафедры Информационных технологий
протокол № 6 от 22.03.2024

Заведующий кафедрой ВМиФ

_____ В.В. Левитес

1. Критерии и средства оценивания компетенций и индикаторов их достижения, формируемых дисциплиной (модулем)

Код и наименование компетенции	Код и наименование индикатора(ов) достижения компетенции	Результаты обучения по дисциплине (модулю)			Оценочные средства текущего контроля	Оценочные средства промежуточной аттестации
		Знать	***Уметь***	***Владеть***		
ОПК-2: Способен использовать и адаптировать существующие математические методы и системы программирования для разработки и реализации алгоритмов решения прикладных задач	ОПК-2.1 Использует и адаптирует существующие математические методы для разработки и реализации алгоритмов решения прикладных задач ОПК-2.2 Использует существующие системы программирования для разработки и реализации алгоритмов решения прикладных задач	– историю развития функционального программирования; – основные свойства ЯФП; – круг задач, решаемых методами ЯФП. – понятие и назначение лямбда-функции; – представление об основных языках функционального программирования – теоретические основы языков логического программирования основы языка Пролог	– определять степень адекватности конкретных задач для их решения с помощью того или иного языка функционального или логического программирования; – определять и вызывать собственные лямбда-функции; – использовать инструментарий стандартных лямбда-функций. использовать язык Пролог для создания запросов в подходящих базах данных.	– методологией написания программ средствами ЯФП и ЯЛП в интегрированных средах разработки. – технологией описания функциональных типов; – технологией работы с классами и объектами на примере суперклассов, собственных классов. – технологией организации ввода/вывода данных; работы с файлами; – технологией построения формальной системы как некоторого формального исчисления.	- комплект заданий для выполнения лабораторных работ; - тестовые задания;	Результаты текущего контроля

2. Оценка уровня сформированности компетенций (индикаторов их достижения)

Показатели оценивания компетенций (индикаторов их достижения)	Шкала и критерии оценки уровня сформированности компетенций (индикаторов их достижения)			
	Ниже порогового *(«неудовлетворительно»)*	**Пороговый** *(«удовлетворительно»)*	**Продвинутый** *(«хорошо»)*	**Высокий** *(«отлично»)*
Полнота знаний	Уровень знаний ниже минимальных требований. Имели место грубые ошибки.	Минимально допустимый уровень знаний. Допущены не грубые ошибки.	Уровень знаний в объёме, соответствующем программе подготовки. Допущены некоторые погрешности.	Уровень знаний в объёме, соответствующем программе подготовки.
Наличие умений	При выполнении стандартных заданий не продемонстрированы основные умения. Имели место грубые ошибки.	Продемонстрированы основные умения. Выполнены типовые задания с не грубыми ошибками. Выполнены все задания, но не в полном объеме (отсутствуют пояснения, неполные выводы)	Продемонстрированы все основные умения. Выполнены все основные задания с некоторыми погрешностями. Выполнены все задания в полном объёме, но некоторые с недочетами.	Продемонстрированы все основные умения. Выполнены все основные и дополнительные задания без ошибок и погрешностей. Задания выполнены в полном объеме без недочетов.
Наличие навыков (владение опытом)	При выполнении стандартных заданий не продемонстрированы базовые навыки. Имели место грубые ошибки.	Имеется минимальный набор навыков для выполнения стандартных заданий с некоторыми недочетами.	Продемонстрированы базовые навыки при выполнении стандартных заданий с некоторыми недочетами.	Продемонстрированы все основные умения. Выполнены все основные и дополнительные задания без ошибок и погрешностей. Продемонстрирован творческий подход к решению нестандартных задач.
Характеристика сформированности компетенции	Компетенции фактически не сформированы. Имеющихся знаний, умений, навыков недостаточно для решения практических (профессиональных) задач. ИЛИ Зачетное количество баллов не набрано согласно установленному диапазону	Сформированность компетенций соответствует минимальным требованиям. Имеющихся знаний, умений, навыков в целом достаточно для решения практических (профессиональных) задач. ИЛИ Набрано зачетное количество баллов согласно установленному диапазону	Сформированность компетенций в целом соответствует требованиям. Имеющихся знаний, умений, навыков достаточно для решения стандартных профессиональных задач. ИЛИ Набрано зачетное количество баллов согласно установленному диапазону	Сформированность компетенций полностью соответствует требованиям. Имеющихся знаний, умений, навыков в полной мере достаточно для решения сложных, в том числе нестандартных, профессиональных задач. ИЛИ Набрано зачетное количество баллов согласно установленному диапазону

3. Критерии и шкала оценивания заданий текущего контроля

3.1. Критерии и шкала оценивания тестирования

Контрольное (экзаменационное) тестирование: балл рассчитывается пропорционально количеству верно решенных дидактически единиц (модулей):

Количество верно решенных ДЕ	0-5
Количество баллов	По 8 баллов за каждую ДЕ

1. Типовые контрольные задания и методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

1. Функциональное программирование относится к:
а) императивному программированию;
б) объектно-ориентированному программированию;
в) декларативному программированию.

2. Декларативный подход в программировании требует:
а) описывать последовательность действий для нахождения результата;
б) описывать свойства требуемого результата;
в) описывать последовательность действий и свойства требуемого результата.

3. К особенностям декларативных языков программирования относятся:
а) выразительность, параллелизм и ориентированность на присущий человеку образ мышления;
б) параллелизм, полиморфизм и наследование;
в) выразительность, инкапсуляция и ориентированность на присущий человеку образ мышления.

4. К языкам функционального программирования относятся:
а) Lisp, Scheme, Miranda, Haskell, O'Caml, F#, Erland.
б) Lisp, Haskell, Prolog, O'Caml, F#, Clean.
в) Lisp, Haskell, F#, C#, C++, Object Pascal.
г) Lisp, Miranda, Haskell, JavaScript, Visual Prolog, O'Caml, F#, Erland.

5. К языкам логического программирования относятся:
а) Haskell, Cofer.
б) Prolog, Visual Prolog.
в) Prolog, F#.

6. Язык функционального программирования F# наследует от языков:
а) Lisp, O'Caml.
б) Lisp, Haskell.
в) O'Caml, Haskell.

7. Тип выражения говорит о том:
а) какие значения могут получиться при вычислении этого выражения.
б) имеет ли выражение какое-нибудь значение или нет.
в) какие значения могут получиться при вычислении этого выражения и получатся ли какие-нибудь значения вообще.

8. Базовый элемент синтаксиса F#:
а) константа;
б) выражение;
в) объект;

г) класс.

9. Функции в F# объявляются конструкцией:
а) <имя функции> (<аргумент функции>) : = <тело функции>;
б) let <имя функции> (<аргумент функции>) = <тело функции>;
в) let <тип функции> <имя функции> (<аргумент функции>) : {<тело функции>};

10. Функция let plus x y = x+y;; будет иметь тип:
а) int -> int;
б) int -> int -> int;
в) int*int -> int.

11. Результат выражения 0.1+1 будет:
а) 1.1;
б) 1;
в) 0.1;
г) ошибка.

12. Каррирование – это прием функционального программирования, который:
а) функцию с более чем одним аргументом интерпретирует как функцию от первого аргумента, возвращающую функцию от оставшихся, при фиксированном значении первого.
б) позволяет выражениям и функциям иметь более одного типа.
в) функцию с более чем одним аргументом интерпретирует как функцию от первого аргумента при его фиксированном значении.

12. Аппликативный порядок применения функции- это порядок, при котором:
а) аргументы функции начинают вычисляться только тогда, когда в них возникает необходимость;
б) аргументы функции полностью вычисляются до ее вызова.

13. Лямбда-выражение λx.x2-x в F# будет записано в виде:
а) let f x = x*x-x;
б) fun x -> x*x-x;
в) let function x -> x*x-x.

14. Задана функция let sum x = x+x;; При нормальном порядке применения функции sum 3+2 следующий шаг будет:
а) sum 5;
б) (3+2) + (3+2);
в) 5+5;
г) 10.

15. Список – это:
а) конечная последовательность элементов одного типа;
б) бесконечная последовательность элементов одного типа;
в) конечная последовательность элементов разных типов.

16. Хвостом списка называется:
а) первый элемент списка;
б) последний элемент списка;
в) все элементы списка кроме первого.

17. Результат сопоставления let x::y::z = [1;2;3;4] будет:
а) x=[1;2], y=3, z=4;
б) x=1, y=2, z=[3;4];
в) x=1, y=2, z=[];
г) x=1, y=2, z=3.

18. Функция конкатенации («склейки») списков [1;2;3] и [4;5] будет записана как:
а) [1;2;3] + [4;5];
б) [1;2;3][4;5];
в) [1;2;3] and [4;5];
г) [1;2;3] @ [4;5].

19. Сложность добавления элемента в начало списка длины n будет составлять:
а) O(1);
б) O(n);
в) $O(n^2)$;
г) O(1+n).

20. Извлечь из массива A, состоящего из первых 6 элементов, подмассив с 3-го по 5-й элемент можно в виде:
а) A.[3..5];
б) A[2,3,4];
в) A.[2..4];
г) [A[3]; A[4]; A[5]].

21. В классических функциональных языках многомерные массивы представляются:
а) списками массивов;
б) списками списков;
в) массивами списков.

22. Подмассиву с 3-го по 5-й элемент массива A присвоить новые значения больше их индекса на 1 можно в виде:
а) A.[2..4] <- [|3..5|];
б) A.[3..5] = [4..6];
в) A[2..4] -> 3..5;
г) A[3..5] = [|4; 5; 6|].

23. Матрицы в библиотеке F# имеют тип:
а) Matrix[ ][ ];
б) Vector<_>;
в) Matric( );
г) Matrix<_>.

Ключ к тестовым заданиям

№ вопроса	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Ответ	В	Б	А	А	Б	В	В	Б	Б	Б	Г	А
№ вопроса	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**-**
Ответ	Б	Б	А	В	Б	Г	А	В	Б	А	Г	-

Примерные вопросы к зачету
1. Парадигма функционального программирования
2. Основные свойства функциональных языков
3. Типовые задачи, решаемые методами функционального программирования
4. Конструирование функций

5. Доказательство свойств функций. Примеры.
6. Списки. Операции со списками. Примеры.
7. Использование функций для описания процессов вычисления.
8. Типизация данных и функций.
9. Модули и абстрактные типы данных.
10. Понятие о полиморфизме. Область применения полиморфизма. Примеры.
11. Понятие о наследование. Примеры реализации.
12. Классы как способ абстракции. Реализация работы с классами на примере одного из ЯФП.
13. Стандартные классы (суперклассы) на примере одного из ЯФП.
14. Понятие монады. Монада как тип-контейнер.
15. Монады. Последовательное выполнение действий.
16. Стандартные монады на примере одного из ЯФП.
17. Разработка собственных монад.
18. Основы комбинаторной логики.
19. Лямбда-исчисление как теоретическая основа функционального программирования.
20. Редукция и вычисления в функциональных языках.
21. Понятие о математической лингвистике.
22. Понятие транслятора. Теория построения трансляторов средствами ЯФП.
23. Возможности библиотек для создания трансляторов.
24. Понятие об искусственном интеллекте. Основные задачи искусственного интеллекта.
25. Перспективы функционального программирования.