**КНУ ім. Тараса Шевченка**

**Факультет інформаційних технологій**

**Кафедра програмних систем і технологій**

# Екзаменаційні запитання з дисципліни «Функціональне програмування» (2021)

1. Поняття, характеристика, особливості парадигм декларативного та імперативного програмування, різниця між цими парадигмами. Приклади програм, що написані мовою імперативного та функціонального програмування
2. Характеристика мов функціонального програмування. Переваги та недоліки мов функціонального програмування. Області застосування мов функціонального програмування. Навести приклади програми функціональною мовою програмування
3. Базові концепції мов функціонального програмування (Scheme, Lisp, Haskell та інш). Поняття програмування, що управляється даними.
4. Функції у функціональному програмуванні: математичне визначення функції та її еквівалентне означення у функціональному програмуванню, застосування (виклик) функції, поняття параметрів та значень функції, поняття прозорості посилань, функція як величина даних першого класу, композиція функцій, поняття функцій вищих порядків.
5. Підстановочна модель застосування процедур мови Scheme: поняття, аплікативний і нормальний порядки обчислення процедур. Навести приклади різних значень процедур в залежності від порядку обчислення процедури.
6. Процедури і процеси, які вони породжують: лінійна рекурсія та ітерація, деревоподібна рекурсія, різниця між рекурсивною процедурою та рекурсивним процесом, хвостова рекурсія (мінімізація неефективності рекурсії), глибина стеку, прийоми перетворення не хвостової рекурсії у хвостову (використання параметра-акумулятора), хвостові виклики та хвостовий контекст. Навести приклади рекурсивних процедур, хвостової рекурсії та переповнення стеку.
7. Інтерпретація поняття змінної в імперативних та функціональних мовах програмування. Альтернатива присвоєнню в строго функціональних мовах. Навести приклади змінних як фактичних значень, змінних як областей пам’яті, прив’язки значень до змінних.
8. Вирази в мові Scheme (Lisp): типи виразів, стандартне правило обчислення, префіксна форма, приклади застосування стандартного правила обчислення виразів до атомів, символів, виразів в дужках, підвиразів.
9. Класифікація типів виразів (примітивні та похідні). Примітивні типи виразів: посилання на змінні, літерні вирази, процедурі виклики, управління обчисленням (моделювання розгалуження (if), моделювання циклів). Навести приклади перерахованих типів виразів.
10. Класифікація типів виразів (примітивні та похідні). Похідні типи виразів: умовні вирази (cond), логічні операції (and, or), конструкти, що пов'язують (let, let\*, letrec), послідовне виконання (begin), ітерації (do), поняття відкладеного обчислення, квазі цитування. Навести приклади перерахованих типів виразів.
11. Булеві типи даних і предикати.
12. λ-числення як основа визначення функцій (процедур). Функції (процедури) вищих порядків: процедури в якості аргументів, побудова процедур за допомогою форми lambda, процедури як значення, що повертаються. Навести приклади коду для застосування процедур вищих порядків.
13. Проста рекурсія. Рекурсія за значенням і рекурсія по аргументу. Паралельна рекурсія. Взаємна рекурсія. Рекурсія більш високого порядку.
14. S-вирази як основні типи даних в мові Лісп. Списки. Рекурсивне визначення списку. Базові функції обробки S-виразів CAR, CDR, CONS, ATOM, EQ, COND
15. Поняття списку як основної структури даних мови Лісп. Графічне представлення облікової структури. Реалізація на базі основних функцій CAR, CDR, CONS інших функцій: Append (злиття двох списків), Length (визначення довжини списку), Reverse (реверсування списку)
16. Накопичуючий параметр - акумулятор. Принципи побудови визначень з накопичуючим параметром. Приклади використання накопичуючого параметра. підставкова модель:
17. Функції SET, SETQ, SETF. Керуючі структури мови Лісп: робота з контекстом, послідовне виконання, розгалуження обчислень, ітерації.
18. Поняття макроса в Ліспі. Порядок обчислення макросу: етапи розширення, обчислення. Визначення нових синтаксичних форм з допомогою макросів. тестування макросів
19. Побудова системи аналітичного диференціювання виразів алгебри
20. Рекурсія. Поняття підстановочної моделі, види рекурсії. Рекурсивний процес, види рекурсивних процесів, ітеративний процес.
21. Числові типи: башта підтипів, поняття точності чисел в Scheme (Lisp), синтаксис числових констант, числові операції (предикати числових типів, предикати кількісної перевірки точності, предикати порівняння, перевірка властивості числа (0, парне, додатне…), арифметичні, max, min, НОД, тригонометричні, … ). Навести приклади коду.
22. Методологія програмування «абстракція даних»: поняття даних, поняття абстракції, переваги, бар’єри абстракції, приклади застосування, роль селекторів і конструкторів для реалізації абстракції даних. Навести приклади коду
23. Типи даних «пара», «список»: визначення типів, нотації (у тому числі точкова нотація), базові процедури для роботи зі списками: побудова списку (list), селектори (car, cdr), конструктори списку (cons), предикати( null?, atom?, list?, pair?) композиція селекторів (caar, cadr, cdddr), стандартні прийоми обробки списків («cdr вниз и cons вверх»). Навести приклад коду для використання пари та списку. Дати графічну інтерпретацію списку та операцій над ним
24. Тип даних «список»: стандартні прийоми обробки списків («cdr вниз и cons вверх»), операції зі списком (кількість елементів списку, інвертування списку, додавання списків, застосування процедури перетворення до кожного елемента списку). Навести приклад коду для використання пари та списку. Дати графічну інтерпретацію списку та операцій над ним.
25. Ієрархічні структури: дерева, бінарні дерева, дерева бінарного пошуку, процедури створення, пошуку, фільтрації даних, перетворення елементів дерева, накопичення перетворених даних. Навести приклад коду.
26. Множини як неупорядковані списки: створення множини, додавання елементів, перевірки співпадання елементів, реалізація множинних операцій (перетин, об’єднання, різниця, приналежність елемента множині). Дати приклади коду.
27. Множини як упорядковані списки: створення множини, додавання елементів, перевірки співпадання елементів, реалізація множинних операцій (перетин, об’єднання, різниця, приналежність елемента множині). Розрахунок ефективності у порівнянні з неупорядкованим списком. Дати приклади коду.
28. Множини як бінарні дерева: створення множини, додавання елементів, перевірки співпадання елементів, реалізація множинних операцій (перетин, об’єднання, різниця, приналежність елемента множині). Дати приклади коду.
29. Лямбда-числення як основа визначення функцій в Scheme (Lisp, Haskell та інш.): математичне означення лямбда-числення, поняття lambda–виразу, формальний синтаксис lambda форми (для створення безіменної процедури), створення локальних змінних (з lambda та let), процес виконання lambda-виразу, lambda–вираз і хвостовий контекст. Навести приклади використання lambda–виразу.
30. Символьні дані: синтаксис, бібліотечні процедури обробки символів, подання алгебраїчних виразів через списки символів, подання даних для задачі символьного диференціювання (подання змінної, суми та добутку)
31. Символьні дані: арифметика многочленів (додавання, віднімання, множення, ділення), побудова списку термів для операцій над многочленами, ієрархія типів в символьній алгебрі.
32. Рядки: визначення рядка, рядкові константи, індекси рядка, стандартні процедури, що оперують рядками (копіювання рядків, порівняння рядків, пошук підрядка в рядку, конкатенація рядків). Навести приклади коду, що обробляє рядки.
33. Вектори: поняття, визначення, порівняння вектора та списку, поняття довжини та індексів вектора, запис векторів, векторні константи, стандартні процедури обробки векторів (предикат перевірки вектора, створення вектора з заданої кількості елементів, довжина вектора, визначення значення заданого елемента вектора). Навести приклади коду роботи з векторами.
34. Присвоєння у функціональних мовах: роль оператора присвоєння в мовах програмування, форма set! для побудови об’єктів, що мають внутрішній стан, моделювання стану програми за допомогою локальних змінних, моделювання зміну стану за допомогою оператора присвоєння. Навести приклад коду застосування оператора присвоєння та зміни стану програми.
35. Модель обчислення з оточеннями та підстановочна обчислювальна модель: поняття, порівняння, правило реалізації моделі обчислення з оточеннями, роль складеної процедури та lambda форми для реалізації моделі обчислення з оточенням. Навести приклади реалізації моделі обчислення з оточеннями та подання об’єктів, що мають внутрішній стан.
36. Списки, що змінюються: поняття мутаторів списків, елементарні мутатори для пар, конструктори з мутаторами для списків, застосування присвоєння для об’єктів, що змінюються. Навести приклади використання мутаторів та присвоєння для списків, що змінюються.
37. Композиції селекторів списків. Конструктори списків.
38. Таблиці: технологія побудова таблиць за допомогою списків, що змінюються, одновимірні таблиці з одним ключем (поняття хребта таблиці, додавання даних в таблицю, видобування даних з таблиці), двовимірні таблиці (використання двох ключів, додавання даних в таблицю, видобування даних з таблиці). Приклади коду роботи з таблицями.
39. Черги: поняття черги, подання черги як структури даних, набір операцій, що визначають чергу (конструктор, селектори, мутатори), неефективність подання черги як звичайного списку, додавання та вилучення елементів з черги. Навести приклади коду для роботи з чергами.
40. Концепція функціонального програмування: відмінності від імперативного програмування, особливості функціонального стилю програмування, приклад коду функціональною та алгоритмічною мовами.
41. Математичні функції як базіс функціонального програмування
42. Основні принципи функціонального програмування
43. Порівняльний аналіз деяких мов функціонального програмування, зокрема сім’я LISP
44. Особливості мови Scheme : конструктивні особливості, області використання мови
45. Механізми програмування Scheme: елементарні вирази, засоби комбінування, засоби абстракції, типи об’єктів
46. Основні типи в мові Scheme (Lisp, Haskell), навести приклади
47. Вирази в мові Scheme: 7 типів виразів, їх особливості, приклади
48. Комбінаційні форми: префіксна нотація, загальне правило обчислення комбінацій, рекурсивне обчислення комбінацій
49. Аплікативний і нормальний порядки обчислення процедур в Scheme (Lisp)
50. Основні правила і приклади побудови рекурсивних функцій, навести приклади.
51. Лямбда-обчислення як основа визначення функцій в мовах функціонального програмування, навести приклади.
52. Відмінності між лінійно-рекурсивним процесом обчислень та лінійно-ітеративний процес обчислення, навести приклади
53. Поняття хвостової рекурсії та її оптимізація в мовах функціонального програмування. Приклад хвостової рекурсії на Scheme (Lisp, Haskell)
54. Процедури вищого порядку в Scheme (Lisp, Haskell): процедури як аргументи, процедури як значення, що повераються. Приклади коду.
55. Поняття абстракції даних в мові Scheme (Lisp): ідея абстракції даних, інтерфейс між абстрактними даними і конкретним уявлення складових даних,
56. Арифметика раціональних чисел в мові Scheme (Lisp) як приклад створення складених даних
57. Складова структура даних – пара, конструктори та селектори пари
58. Ієрархічні дані та властивість замикання. Навести приклади коду
59. Поняття послідовності в мові Scheme (Lisp) як впорядкованої сукупності об’єктів даних: подання послідовностей, стрілочна діаграма як спосіб графічного зображення послідовності
60. Точкові пари в мові Scheme (Lisp): області застосування, відмінність від спискового запису, приклади коду
61. Типові операції обробки списків чк послідовностей об’єктів, переваги використання списків.
62. Вектори, рядки, дерева як різновиди спискових структур в мові Scheme (Lisp): відмінності цих структур одна від одної.
63. Подання дерев в мові Scheme (Lisp): поняття дерева, абстракція для роботи з деревами (конструктор та селектори дерева).
64. Таблиці в мові Scheme як список записів: подання таблиць із застосуванням структур типу пари
65. Обробка файлів в Scheme (Lisp, Haskell): призначення та поняття портів та іх зв’язок із файлами, технологія обробки файлів.
66. Символьні обчислення в Scheme (Lisp): символьне диференціювання алгебричних виразів.

# Джерела

1. Харольд Абельсон, Джеральд Джей Сассман. Структура и интерпретация компьютерных программ. Добросвет, 2006
2. Ричард Келси, Уильям Клингер и Джонатан Рис. Исправленный отчёт по Алгоритмическому Языку Scheme. 2006 (R6RS Report on the Algorithmic Language SCHEME)
3. Майлингова О.Л., Манжелей С.Г., Соловская Л.Б. Прототипирование программ на языке Scheme (методическое пособие по практикуму). МГУ им. М.В.Ломоносова, 2001
4. Орлов С. А.Теория и практика языков программирования: Учебник для вузов. — СПб.: Питер, 2014. (глава 20)