Затверджено

Наказ Міністерства освіти і науки,

молоді та спорту України

29 березня 2012 р. №384

Форма№Н-5.05

Київський національний університет ім. Тараса Шевченка

|  |  |  |
| --- | --- | --- |
| **ОКР** |  | бакалавр |
| **Спеціальність** |  | 121 Інженерія програмного забезпечення |
| **Семестр** |  | сьомий |
| **Навчальна дисципліна** |  | Функціональне програмування |

# ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ № 32

1. **Теоретичне запитання (5 балів).**

λ-числення як основа визначення функцій (процедур) в функціональному програмуванні. 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.

1. **Написати програму мовою функціонального програмування (20 балів)**

Створити чергу з алфавітно-цифрових символів. Побудувати стек з тих елементів черги, які є голосними латинськими літерами. Цифрові символи з черги записати в окремий список. Вивести на екран побудовані стек, список та кількість елементів в них

1. **Виконати Тест1 (Scheme+Lisp) або Test2 (Haskell+Scheme) на вибір студентв, що подані в Google Forms за посиланням (10 балів):**

**Тест1 (Scheme+Lisp):**

https://docs.google.com/forms/d/e/.....

**Test2 (Haskell+Scheme):**

[https://docs.google.com/forms/d/e/](https://docs.google.com/forms/d/e/1FAIpQLSfCxpOpST5ugw2bt5shUEz-YUY5MCZjq3EQMKDzxUV4uFFYHw/viewform).......

1. **Пояснити семантику поданого нижче коду (на вибір студента код мовою SCHEME або HASKELL), записавши коментарі в рядках коду, відредагувати, дописати код у випадку необхідності, скомпілювати та виконати його. Проінтерпретувати отримані результати. (5 балів)**

|  |  |
| --- | --- |
| SCHEME | HASKELL |
| (infix '(+ a (\* b c))) => (a + (b \* c))  (define (infix x)  (if (pair? x)  (if (pair? (cddr x)) ; test if binary  (list (infix (cadr x)) ; binary  (car x)  (infix (caddr x)))  (list (car x) (infix (cadr x)) ) ) ; unary  x)) | import Data.List ( sort )  main :: IO ()  main = do  putStrLn "Enter amount of elements: "  n' <- getLine  let n = read n' :: Int  let list\_input = take n [(-n) `div` 2.. n `div` 2]  print list\_input  let list1 = [if x < 0 then x^2 else x | x <- list\_input]  print list1  let list2 = reverse (sort list1)  print list2  let sum3 = sum [x | x <- list2, even x]  putStrLn "\nSum of numbers: "  print sum3 |

Затверджено на засіданні кафедри програмних систем і технологій, протокол №2 від «11» вересня 2021р.

Завідувач кафедри Бичков О.С.

Екзаменатор Ковалюк Т.В.