Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Уфимский университет науки и технологий»

Факультет информатики и робототехники

Кафедра вычислительной математики и кибернетики

Отчет по лабораторной работе №2

По дисциплине «Функциональное программирование»

Выполнил:

студент группы МО-321 Шемануев А.Е.

Проверил:

Ризванов Д. А.

Уфа, 2022

**Задание 1**

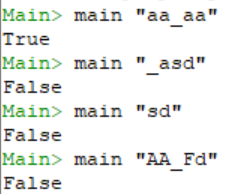
В задании описаны правила составления некоторого пароля. Написать

функцию, проверяющую удовлетворяет ли заданная строка данным

правилам.

а)должен содержаться знак подчеркивания, но не на первом и не на последнем месте; б)количество заглавных букв не должно превышать количество строчных.

|  |
| --- |
| import Data.Char  checkA :: String -> Bool  checkB :: String -> Bool  countUpper :: String -> Int  countLower :: String -> Int  countUpper [] = 0  checkA str = (elem '\_' str) && not(last str == '\_') && not(head str == '\_')  checkB str = (length str - countUpper str) >= (countUpper str)  main a = (checkA a) && (checkB a) |

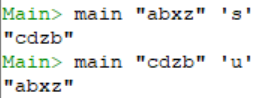


**Задание 2**

Зашифруйте текст данного задания с помощью шифра Цезаря с ключом,

соответствующим номеру варианта.

|  |
| --- |
| import Data.Char  shifr :: Char -> Char  unshifr :: Char -> Char  shifr c | ((elem (ord c + 2) [65..90]) || (elem (ord c + 2) [97..122])) = chr (ord c + 2)  | ((elem (ord c + 2) [91..92]) || (elem (ord c + 2) [123..124])) = chr (ord c + 2 - 26)  | True = c  unshifr c | ((elem (ord c - 2) [65..90]) || (elem (ord c - 2) [97..122])) = chr (ord c - 2)  | ((elem (ord c - 2) [63..64]) || (elem (ord c - 2) [95..96])) = chr (ord c - 2 + 26)  | True = c  main str c | c == 's' = map shifr str  | c == 'u' = map unshifr str  | True = "No" |



**Задание 3**

Написать функцию, вычисляющую значение выражения и выводящую

ответ в десятичном представлении, функция должна иметь не более двух

аргументов строкового типа.

5248 +CCXCI

|  |
| --- |
| import Data.Char(digitToInt)  conver8\_10 :: String -> Int  converX\_10 :: String -> Int  converX\_10\_ :: String -> Int  -- 1 - I, 5 - V, 10 - X, 50 - L, 100 - C, 500 - D и 1000 - M.  mySum :: String -> String-> Int  getBefore :: String -> String  getAfter :: String -> String  smartSum :: String -> Int  rimToInt :: Char -> Int  rimToInt c | c == 'I' = 1  | c == 'V' = 5  | c == 'X' = 10  | c == 'L' = 50  | c == 'C' = 100  | c == 'D' = 500  | c == 'M' = 1000  | True = error "SMTH go wrong"  conver8\_10 "" = 0  conver8\_10 (h:st) = (digitToInt h) \* (8 ^ length st) + (conver8\_10 st)  converX\_10 "" = 0  converX\_10 (a:(h:st)) | (rimToInt a) < (rimToInt h) = (converX\_10 (h:st)) - (rimToInt a)  | True = (converX\_10 (h:st)) + (rimToInt a)  converX\_10 (h:st) = (converX\_10 st) + (rimToInt h)  converX\_10\_ (a:(h:st)) = (rimToInt h) + (converX\_10 st)  mySum i1 i2 = (conver8\_10 i1) + (converX\_10 i2)  getBefore (h:st) | h == '+' = ""  | True = (h:getBefore st)    getAfter ('+':st) = st  getAfter (h:st) = getAfter st  smartSum s1 = mySum (filter (\xs -> (xs /=' ')) (getBefore s1)) (filter (\xs -> (xs /=' ')) (getAfter s1))  main i1 = smartSum i1 |



**Задание 4**

Дано универсальное множество и некоторые его подмножества A,B,C. Найти множество D, заданное в варианте.



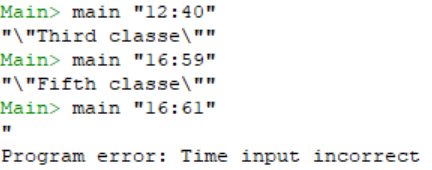
|  |
| --- |
| not\_ :: [Int] -> [Int]  or\_ :: [Int] -> [Int] -> [Int]  tri\_ :: [Int] -> [Int] -> [Int]  and\_ :: [Int] -> [Int] -> [Int]  del\_ :: [Int] -> [Int] -> [Int]  not\_ mas = filter (\x -> not(elem x mas)) [1..10]  or\_ mas1 mas2 = filter (\x -> (elem x mas1) || (elem x mas2)) [1..10]  and\_ mas1 mas2 = filter (\x -> (elem x mas1) && (elem x mas2)) [1..10]  tri\_ mas1 mas2 = or\_ (del\_ mas1 mas2) (del\_ mas2 mas1)  del\_ mas1 mas2 = filter (\x -> (elem x mas1) && not(elem x mas2)) [1..10]  main a b c = tri\_ a (or\_ (not\_ b) c) |



**Задание 5**

По введенной строке формата «hh:mm» печатает сообщение о номере пары или перерыва в УГАТУ, соответствующих данному времени (10:21 – вторая пара; 15:55 – второй большой перерыв и т.п.). При введенных значениях вне учебного времени (либо не являющихся временем – 52:81 например) выводит соответствующие сообщения об ошибках.

|  |
| --- |
| import Data.Char(digitToInt)  getBefore :: String -> String  getAfter :: String -> String  getBefore (h:st) | h == ':' = ""  | True = (h:getBefore st)  getAfter (':':st) = st  getAfter (h:st) = getAfter st  hmIntoMin :: String -> Int  hmIntoMin i | ((read (getBefore i)) > 23) || ((read (getBefore i)) < 0) || ((read (getAfter i)) > 59) || ((read (getAfter i)) < 0) = error "Time input incorrect"  | True = (read (getBefore i)) \* 60 + (read (getAfter i))  getWhatToDo i | elem i [480..574] = show "First classe"  | elem i [575..584] = show "First small break"  | elem i [585..679] = show "Second classe"  | elem i [680..729] = show "First big break"  | elem i [730..824] = show "Third classe"  | elem i [825..834] = show "Second small break"  | elem i [835..929] = show "Fourth classe"  | elem i [930..969] = show "Second big break"  | elem i [970..1064] = show "Fifth classe"  | elem i [1065..1074] = show "Third small break"  | elem i [1075..1169] = show "Sixth classe"  | elem i [1170..1179] = show "Fourth small break"  | elem i [1180..1269] = show "seventh classe"  | True = error "Time outside the school day"    main i = getWhatToDo (hmIntoMin i) |



**Задание 6**

Подключите модуль Trace к модулю, в котором определены функции из

задания 3 Запросите в интерпретаторе, объясните тип и проверьте поведение

на примерах входных данных для функции trace.

|  |
| --- |
| import Debug.Trace  import Data.Char(digitToInt)  conver8\_10 :: String -> Int  converX\_10 :: String -> Int  converX\_10\_ :: String -> Int  mySum :: String -> String-> Int  getBefore :: String -> String  getAfter :: String -> String  smartSum :: String -> Int  rimToInt :: Char -> Int  rimToInt c | c == 'I' = 1  | c == 'V' = 5  | c == 'X' = 10  | c == 'L' = 50  | c == 'C' = 100  | c == 'D' = 500  | c == 'M' = 1000  | True = error "SMTH go wrong"    conver8\_10 "" = trace "conver8\_10 \"\"" 0  conver8\_10 (h:st) = trace "conver8\_10 (h:st)" (digitToInt h) \* (8 ^ length st) + (conver8\_10 st)  converX\_10 "" = trace "converX\_10 \"\"" 0  converX\_10 (a:(h:st)) | (rimToInt a) < (rimToInt h) = trace "converX\_10 (a:(h:st)) 1" (converX\_10 (h:st)) - (rimToInt a)  | True = trace "converX\_10 (a:(h:st)) 2" (converX\_10 (h:st)) + (rimToInt a)    converX\_10 (h:st) = trace "converX\_10" (converX\_10 st) + (rimToInt h)    converX\_10\_ (a:(h:st)) = trace "converX\_10\_ (a:(h:st))" (rimToInt h) + (converX\_10 st)  mySum i1 i2 = trace "mySum i1 i2" (conver8\_10 i1) + (converX\_10 i2)  getBefore (h:st) | h == '+' = trace "getBefore (h:st) 1" ""  | True = trace "getBefore (h:st) 2" (h:getBefore st)    getAfter ('+':st) = trace "getAfter ('+':st)" stgetAfter (h:st) = trace "getAfter (h:st)" getAfter st  smartSum s1 =trace "smartSum s1" mySum (filter (\xs -> (xs /=' ')) (getBefore s1)) (filter (\xs -> (xs /=' ')) (getAfter s1))  main i1 = trace "123 " (smartSum i1) |

