**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИИ**  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**  
**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №3

Функциональное и логическое программирование

Выполнил: ст. группы ПВ-21  
Ковалев Павел Александрович

Проверил: Поляков В.К.

**Белгород 2020**

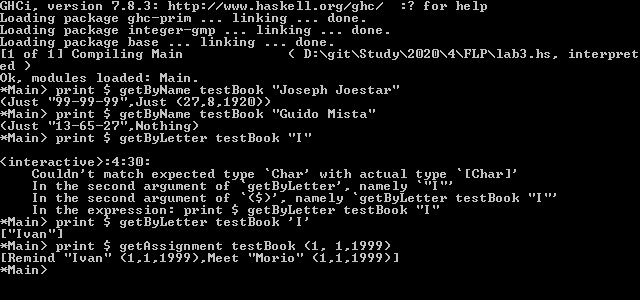
**Вариант 8**

**Задания**

В электронной записной книжке хранятся записи следующих видов: напоминания о днях рождения знакомых, телефоны знакомых и назначенные встречи. Напоминание состоит из имени знакомого и даты. Запись о телефоне должна содержать имя человека и его телефон. Информация о назначенной встече содержит дату встречи и краткое описание. Разработайте тип данных, представляющий такую запись. Записная книжка является списком записей. Определите следующие функции:

1)getByName, возвращающая информацию о человеке с указанным именем (его телефон и дату рождения).

2)getByLetter, возвращающая список людей, о которых есть информация в записной книжке и чье имя начинается на указанную букву.

3)getAssignment, возвращающая по указанной дате список дел (информация о назначенных встречах и телефоны друзей, которых нужно поздравить в этот день).

**Код программы**

type Name = String

type Phone = String

type Descriptions = String

type Date = (Int,Int,Int)

data Records = Remind Name Date |

PhoneBook Name Phone |

Meet Descriptions Date

deriving Show

getByName:: [Records] -> String -> (Maybe Phone, Maybe Date )

getByName l p = (getPhone l, getRemind l)

where getPhone [] = Nothing

getPhone ( PhoneBook p' n :xs)

| p' == p = Just n

| otherwise = getPhone xs

getPhone (\_:xs) = getPhone xs

getRemind [] = Nothing

getRemind ( Remind p' d :xs)

| p' == p = Just d

| otherwise = getRemind xs

getRemind (\_:xs) = getRemind xs

unical n l

| elem n l = l

| otherwise = n:l

getByLetter:: [Records] -> Char -> [Name]

getByLetter [] \_ = []

getByLetter ( PhoneBook [] \_ :xs) c = getByLetter xs c

getByLetter ( PhoneBook p@(n:\_) \_ :xs) c

| n == c = unical p (getByLetter xs c)

| otherwise = getByLetter xs c

getByLetter ( Remind [] \_ :xs) c = getByLetter xs c

getByLetter ( Remind p@(n:\_) \_ :xs) c

| n == c = unical p (getByLetter xs c)

| otherwise = getByLetter xs c

getByLetter (\_:xs) c = getByLetter xs c

getAssignment:: [Records] -> Date -> [Records]

getAssignment [] \_ = []

getAssignment (r@(Remind \_ d'):xs) d

| d == d' = r : getAssignment xs d

| otherwise = getAssignment xs d

getAssignment (r@(Meet \_ d'):xs) d

| d == d' = r : getAssignment xs d

| otherwise = getAssignment xs d

getAssignment (\_:xs) d = getAssignment xs d

testBook = [Remind "Joseph Joestar" (27, 8,1920),

Remind "Ivan" (1, 1,1999),

PhoneBook "Joseph Joestar" "99-99-99",

PhoneBook "Guido Mista" "13-65-27",

Meet "Morio" (1, 1,1999)]