Міністерство освіти і науки України
Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Програмування інтелектуальних інформаційних систем

ЗВІТ

до лабораторних робіт

Виконав студент ІП-01 Хернуф Валід _____
(№ групи, прізвище, ім'я, по батькові)

Прийняв ас. Очеретяний О. К. _____
(посада, прізвище, ім'я, по батькові)

Київ 2022

1. Завдання лабораторної роботи

Написати 11 функцій SML (і тести для них), пов’язаних з календарними датами. У всіх завданнях, **“дата”** є значенням SML типу int*int*int, де перша частина - це рік, друга частина - місяць і третя частина - день. **«Правильна»** дата має позитивний рік, місяць від 1 до 12 і день не більше 31 (або 28, 30 - залежно від місяця). Перевіряти “правильність” дати не обов'язково, адже це досить складна задача, тож будьте готові до того, що багато ваших функцій будуть працювати корректно для деяких/всіх **“неправильних”** дат у тому числі. Також, **«День року»** — це число від 1 до 365 де, наприклад, 33 означає 2 лютого.

1. Напишіть функцію is_older, яка приймає дві дати та повертає значення true або false. Оцінюється як true, якщо перший аргумент - це дата, яка раніша за другий аргумент. (Якщо дві дати однакові, результат хибний.)

2. Напишіть функцію number_in_month, яка приймає список дат і місяць (тобто int) і повертає скільки дат у списку в даному місяці.

3. Напишіть функцію number_in_months, яка приймає список дат і список місяців (тобто список int) і повертає кількість дат у списку дат, які знаходяться в будь-якому з місяців у списку місяців. **Припустимо, що в списку місяців немає повторюваних номерів**. Підказка: скористайтеся відповіддю до попередньої задачі.

4. Напишіть функцію dates_in_month, яка приймає список дат і число місяця (тобто int) і повертає список, що містить дати з аргументу “список дат”, які знаходяться в переданому місяці. Повернутий список повинен містять дати в тому порядку, в якому вони були надані спочатку.

5. Напишіть функцію dates_in_months, яка приймає список дат і список місяців (тобто список int) і повертає список, що містить дати зі списку аргументів дат, які знаходяться в будь-якому з місяців у списку місяців. Для простоти, припустимо, що в списку місяців немає повторюваних номерів. Підказка: Використовуйте свою відповідь на попередню задачу та оператор додавання списку SML (@).

6. Напишіть функцію get_nth, яка приймає список рядків і int n та повертає n-й елемент списку, де голова списку є першим значенням. Не турбуйтеся якщо в списку занадто мало елементів: у цьому випадку ваша

функція може навіть застосувати hd або tl до порожнього списку, і це нормально.

7. Напишіть функцію date_to_string, яка приймає дату і повертає рядок у вигляді “February 28, 2022” Використовуйте оператор ^ для конкатенації рядків і бібліотечну функцію Int.toString для перетворення int в рядок. Для створення частини з місяцем не використовуйте купу розгалужень. Замість цього використайте список із 12 рядків і свою відповідь на попередню задачу. Для консистенції пишіть кому після дня та використовуйте назви місяців англійською мовою з великої літери.

8. Напишіть функцію number_before_reaching_sum, яка приймає додатний int під назвою sum, та список int, усі числа якої також додатні. Функція повертає int. Ви повинні повернути значення int n таке, щоб перші n елементів списку в сумі будуть менші sum, але сума значень від n + 1 елемента списку до кінця був більше або рівний sum.

9. Напишіть функцію what_month, яка приймає день року (тобто int між 1 і 365) і повертає в якому місяці цей день (1 для січня, 2 для лютого тощо). Використовуйте список, що містить 12 цілих чисел і вашу відповідь на попередню задачу.

10. Напишіть функцію month_range, яка приймає два дні року day1 і day2 і повертає список int [m1,m2,...,mn] де m1 – місяць day1, m2 – місяць day1+1, ..., а mn – місяць day2. Зверніть увагу, що результат матиме довжину day2 - day1 + 1 або довжину 0, якщо day1>day2.

11. Напишіть найстарішу функцію, яка бере список дат і оцінює параметр (int*int*int). Він має оцінюватися як NONE, якщо список не містить дат, і SOME d, якщо дата d є найстарішою датою у списку.

2. Опис програмного коду

Функція 1:

```
fun is_older ((y1,m1,d1),(y2,m2,d2))=
    if y1 > y2 then false else if y2 > y1 then true
    else if m1 > m2 then false else if m2 > m1 then true
    else if d1 < d2 then true else false
```

Функція 2:

```
fun number_in_month (dateList : (int*int*int) list, month : int) =
    if null dateList
    then 0
    else if (#2 (hd dateList)) = month
    then 1 + number_in_month (tl dateList, month)
    else number_in_month(tl dateList, month)

(*Also second task. Just alternative*)
fun number_in_month2 ([], _) = 0
  | number_in_month2 ((_,m,_)::dates,month) =
    if m = month then 1 + number_in_month2(dates, month)
    else number_in_month2(dates, month)
```

Функція 3:

```
fun number_in_months (_, []) = []
  | number_in_months (dateList : (int*int*int) list, month::monthes) =
    number_in_month2(dateList, month) :: number_in_months(dateList, monthes)
```

Функція 4:

```
fun dates_in_month ([], _) = []
  | dates_in_month (dateList : (int*int*int) list, month : int) =
    if (#2 (hd dateList)) = month then (hd dateList) :: dates_in_month((tl
dateList), month)
    else dates_in_month((tl dateList), month)
```

Функція 5:

```
fun dates_in_months (_, []) = []
  | dates_in_months (dateList, month::monthes) =
    dates_in_month(dateList, month) @ dates_in_months(dateList, monthes)
```

Функція 6:

```
fun get_nth (strList : string list, num : int) =
    let fun get_nth_count ([], _) = ""
          | get_nth_count (str::strs, from : int) =
            if from = num
            then str
            else get_nth_count(strs, from + 1)
    in get_nth_count(strList, 1) end
```

Функція 7:

```
fun date_to_string (date : int*int*int) =
    let val month_str = ["January",
        "February", "March", "April",
        "May", "June", "July", "August",
        "September", "November", "October",
        "December"] : string list
        fun get_str_month ([], _) = ""
          | get_str_month (month::monthes, cnt : int) =
            if (#2 date) = cnt then month
            else get_str_month(monthes, cnt + 1)
    in
        get_str_month(month_str, 1) ^ " " ^ Int.toString (#3 date) ^ ", " ^
Int.toString (#1 date)
    end
```

Функція 8:

```
fun number_before_reaching_sum (_, []) = 1
  | number_before_reaching_sum (sum, number::numbers) =
    if (sum - number > 0)
    then number_before_reaching_sum(sum - number, numbers) + 1
    else 1
```

Функція 9:

```
fun what_month (day : int) =
    let val month_days = [31, 28,31,30,
        31,30,31,31,30,31,30,31] : int list
    in number_before_reaching_sum(day, month_days) end
```

Функція 10:

```
fun month_range (day1 : int, day2 : int) =
    if day1 > day2
    then []
    else what_month(day1)::month_range(day1 + 1, day2)
```

Функція 11:

```
fun oldest ([]) = NONE
  | oldest (date::dates) =
    let fun older ([], oldestDate) = SOME oldestDate
    | older (date::dates, oldestDate) =
        if is_older(oldestDate, date)
        then older(dates, oldestDate)
        else older(dates, date)
    in older(dates, date) end
```

3. Скріншоти роботи функцій

Тестування функції 1:

```
(*Test for first task*)
fun provided_test1 () =
    let val date1 = (2022,6,21)
        val date2 = (2022,6,20)
        val date3 = (2020,6,20)
        val date4 = (2020,7,20)
    in
    (is_older (date1, date2),
    is_older (date2, date1),
    is_older (date1, date2),
    is_older (date1, date1),
    is_older (date3, date4),
    is_older (date2, date3))
    end


val ans_first = provided_test1()
```

Результат тестування функції 1:

```
val ans_first = (false,true,false,false,true,false) :
  bool * bool * bool * bool * bool * bool
```

Тестування функції 2:

```
fun provided_test2 () =
    let val dateList1 = [(2022,4,3),(2022,2,12),(2021,2,1)] : (int*int*int) list
        val dateList2 = [(2010,2,1),(2015,2,5),(2020,2,10),(2025,2,3)] :
(int*int*int) list
    in
    (number_in_month(dateList1, 4),
    number_in_month(dateList1, 2),
    number_in_month(dateList1, 10),
    number_in_month(dateList2, 2),
    number_in_month2(dateList1, 4),
    number_in_month2(dateList1, 2),
    number_in_month2(dateList1, 10),
```

```
    number_in_month2(dateList2, 2))
    end


val ans_second = provided_test2()
```

Результат тестування функції 2:

```
val ans_second = (1,2,0,4,1,2,0,4) :
  int * int * int * int * int * int * int * int
```

Тестування функції 3:

```
fun provided_test3 () =
    let val dateList1 = [(3,4,2022),(12,2,2022),(1,2,2021)] : (int*int*int) list
        val dateList2 = [(1,2,2010),(5,2,2015),(10,2,2020),(3,2,2025)] :
(int*int*int) list
        val monthList = [4,10,2] : int list
    in
    (number_in_months(dateList1, monthList),
    number_in_months(dateList2, monthList))
    end

val ans_third = provided_test3()
```

Результат тестування функції 3:

```
val ans_third = ([1,0,2],[0,0,4]) : int list * int list
```

Тестування функції 4:

```
fun provided_test4 () =
    let val dateList1 = [(2022,4,3),(2022,2,12),(2021,2,1)] : (int*int*int) list
        val dateList2 = [(2010,2,1),(2015,2,5),(2020,2,10),(2025,2,3)] :
(int*int*int) list
    in
    (dates_in_month(dateList1, 4),
    dates_in_month(dateList1, 2),
    dates_in_month(dateList2, 2),
    dates_in_month(dateList2, 10))
    end


val ans_fourth = provided_test4()
```

Результат тестування функції 4:

```
val ans_fourth =
  ([(2022,4,3)],[(2022,2,12),(2021,2,1)],
   [(2010,2,1),(2015,2,5),(2020,2,10),(2025,2,3)],[]) :
  (int * int * int) list * (int * int * int) list * (int * int * int) list
   * (int * int * int) list
```

Тестування функції 5:

```
fun provided_test5 () =
    let val dateList1 = [(2022,4,3),(2022,2,12),(2021,2,1)] : (int*int*int) list
        val dateList2 = [(2010,2,1),(2015,2,5),(2020,2,10),(2025,2,3)] :
(int*int*int) list
        val monthList1 = [4,10,2] : int list
        val monthList2 = [5,4,3] : int list
    in
    (dates_in_months(dateList1, monthList1),
    dates_in_months(dateList1, monthList2),
    dates_in_months(dateList2, monthList1),
    dates_in_months(dateList2, monthList2))
    end


val ans_fifth = provided_test5()
```

Результат тестування функції 5:

```
val ans_fifth =
  ([(2022,4,3),(2022,2,12),(2021,2,1)],[(2022,4,3)],
   [(2010,2,1),(2015,2,5),(2020,2,10),(2025,2,3)],[]) :
  (int * int * int) list * (int * int * int) list * (int * int * int) list
   * (int * int * int) list
```

Тестування функції 6:

```
fun provided_test6 () =
    let val strList = ["first","second","third","fourth","fifth"] : string list
    in
    (get_nth(strList, 4),
    get_nth(strList, 2),
    get_nth(strList, 1),
    get_nth(strList, 6))
    end


val ans_sixth = provided_test6()
```

Результат тестування функції 6:

```
val ans_sixth = ("fourth","second","first","") :
  string * string * string * string
```

Тестування функції 7:

```
fun provided_test7 () =
    let val date1 = (2022,3,21)
        val date2 = (2021,8,8)
        val date3 = (2014,12,30)
        val date4 = (2005,1,18)
    in
    (date_to_string(date1),
    date_to_string(date2),
    date_to_string(date3),
    date_to_string(date4))
    end


val ans_seventh = provided_test7()
```

Результат тестування функції 7:

```
val ans_seventh =
  ("March 21, 2022","August 8, 2021","December 30, 2014","January 18, 2005") :
  string * string * string * string
```

Тестування функції 8:

```
fun provided_test8 () =
    let val list1 = [12,44,12,67,2] : int list
        val list2 = [1,2,3,4,5,6] : int list
        val list3 = [1,1,1,2,2,1,1,1,1] : int list
    in
    (number_before_reaching_sum(58, list1),
    number_before_reaching_sum(10, list2),
    number_before_reaching_sum(4, list3))
    end

val ans_ninth = provided_test9()
```

Результат тестування функції 8:

```
val ans_eighth = (3,4,4) : int * int * int
```

Тестування функції 9:

```
fun provided_test9() =
    let val day1 = 51 : int
        val day2 = 365 : int
        val day3 = 1 : int
    in
    (what_month(day1),
    what_month(day2),
    what_month(day3))
    end


val ans_ninth = provided_test9()
```

Результат тестування функції 9:

```
val ans_ninth = (2,12,1) : int * int * int
```

Тестування функції 10:

```
fun provided_test10() =
    let val day1 = 31 : int
        val day2 = 32 : int
        val day3 = 365 : int
        val day4 = 330 : int
    in
    (month_range(day1, day2),
    month_range(day3, day4),
    month_range(day4, day3))
    end
```

Результат тестування функції 10:

```
val ans_tenth =
  ([1,2],[],[11,11,11,11,11,12,12,12,12,12,12,12,12,12,12,12,...]) :
  int list * int list * int list
```

Тестування функції 11:

```
fun provided_test11() =
    let val dateList1 = [(2020,12,10),(2022,1,5),(2015,7,1),(2022,1,4)] :
(int*int*int) list
        val dateList2 = [(2022,10,23),(2022,11,1),(2022,1,2)] : (int*int*int)
list
        val dateList3 = [] : (int*int*int) list
    in
    (oldest(dateList1),
    oldest(dateList2),
    oldest(dateList3))
    end
```

Результат тестування функції 11:

```
val ans_eleventh = (SOME (2015,7,1),SOME (2022,1,2),NONE) :
  (int * int * int) option * (int * int * int) option
   * (int * int * int) option
```