Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені ІгоряСікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 2 з дисципліни
«Мультипарадигменне програмування»

„Функціональне програмування”

Виконав(ла) *ІП-01 Адамчук Антон* _____
(шифр, прізвище, ім'я, по батькові)

Перевірив *Очеретяний О. К.* _____
(прізвище, ім'я, по батькові)

Київ 2022

1 ЗАВДАННЯ

Ви напишете 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. (Ми ігноруємо високосні роки, за винятком однієї задачі.)

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 ВИКОНАННЯ

2.1 Програмна реалізація функцій поставлених задач на мові SML

```
fun is_older(date1:int*int*int, date2:int*int*int) =
    if date1 = date2
        then false
    else if (#1 date1) < (#1 date2)
        then true
    else if (#1 date1) = (#1 date2) andalso (#2 date1) < (#2 date2)
        then true
    else if (#1 date1) = (#1 date2) andalso (#2 date1) = (#2 date2) andalso (#3
date1) < (#3 date2)
        then true
    else false;

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

fun number_in_months(dates : (int*int*int) list, monthes : int list) =
    if null monthes
        then 0
    else number_in_month(dates, (hd monthes)) + number_in_months(dates, (tl
monthes));

fun dates_in_month(dates : (int*int*int) list, month : int) =
    if null dates
        then []
    else if ((#2 (hd dates)) = month)
        then (hd dates) :: dates_in_month(tl dates,month)
    else dates_in_month(tl dates,month);

fun dates_in_months (dates : (int*int*int) list, monthes : int list) =
    if null monthes
        then []
    else dates_in_month(dates, (hd monthes)) @ dates_in_months(dates, (tl
monthes));

fun get_nth (text_lines : string list, n:int) =
    if null text_lines
        then ""
    else if n = 1
        then (hd text_lines)
    else get_nth(tl text_lines, n-1);
```

```
fun date_to_string(date : int*int*int) =
    let
        val monthes : string list = ["January", "February", "March", "April",
"May", "June", "July", "August", "September", "October", "November", "December"]
    in
        get_nth(monthes,(#2 date)) ^ " " ^Int.toString(#3 date) ^ ", " ^
Int.toString(#1 date)
    end;

fun number_before_reaching_sum (sum : int, numbers : int list) =
    if null numbers
        then 0
    else if (sum-(hd numbers)<=0)
        then 0
    else number_before_reaching_sum(sum - (hd numbers), (tl numbers))+1;

fun what_month (day : int) =
    let
        val monthes_day = [31,28,31,30,31,30,31,31,30,31,30,31]
    in
        number_before_reaching_sum(day, monthes_day)+1
    end;

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

fun oldest(dates : (int*int*int) list) =
    if null dates
        then "NONE"
    else
    let
        fun greatest(dates : (int*int*int) list) =
            if (null (tl dates))
                then (hd dates)
            else if (is_older((hd dates),greatest(tl dates)))
                then greatest(tl dates)
                else (hd dates)
    in
        "SOME " ^ date_to_string(greatest(dates))
    end;
```

3 ТЕСТУВАННЯ

3.1 Приклади роботи функцій для різних вхідних даних

3.1.1 Функція is_older:

Вхідні дані:

```
is_older((2000,1,15),(2000,1,15)); (*false*)
is_older((1998,1,15),(1998,1,14)); (*false*)
is_older((1998,2,14),(1998,1,14)); (*false*)
is_older((1999,2,14),(1998,1,14)); (*false*)
is_older((1999,1,15),(1999,1,16)); (*true*)
is_older((1999,1,15),(1999,2,15)); (*true*)
is_older((1999,1,15),(2000,1,15)); (*true*)
```

Вихідні дані:

```
[opening hwtask.sml]
val is_older = fn : (int * int * int) * (int * int * int) -> bool
val it = false : bool
val it = false : bool
val it = false : bool
val it = false : bool
val it = true : bool
val it = true : bool
val it = true : bool
```

3.1.2 Функція number_in_month:

Вхідні дані:

```
number_in_month([],2); (*0*)
number_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],4); (*0*)
number_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],1); (*3*)
number_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],3); (*1*)
number_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],2); (*2*)
```

Вихідні дані:

```
val number_in_month = fn : (int * int * int) list * int -> int
val it = 0 : int
val it = 0 : int
val it = 3 : int
val it = 1 : int
val it = 2 : int
```

3.1.3 Функція number_in_months:

Вхідні дані:

```
number_in_months([],[1,2]); (*0*)
number_in_months([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],[]); (*0*)
number_in_months([],[]); (*0*)
number_in_months([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],[1,2]); (*5*)
```

Вихідні дані:

```
val number_in_months = fn : (int * int * int) list * int list -> int
val it = 0 : int
val it = 0 : int
val it = 0 : int
val it = 5 : int
```

3.1.4 Функція dates_in_month:

Вхідні дані:

```
dates_in_month([],1);
dates_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],1);
dates_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],3);
dates_in_month([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],5);
```

Вихідні дані:

```
val dates_in_month = fn :
  (int * int * int) list * int -> (int * int * int) list
val it = [] : (int * int * int) list
val it = [(2000,1,1),(2000,1,2),(2000,1,3)] : (int * int * int) list
val it = [(2000,3,1)] : (int * int * int) list
val it = [] : (int * int * int) list
```

3.1.5 Функція dates_in_months:

Вхідні дані:

```
dates_in_months([],[1,3]);
dates_in_months([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],[]);
dates_in_months([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],[1,3]);
dates_in_months([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2),
(2000,3,1)],[1,3,5]);
```

Вихідні дані:

```
val dates_in_months = fn :
  (int * int * int) list * int list -> (int * int * int) list
val it = [] : (int * int * int) list
val it = [] : (int * int * int) list
val it = [(2000,1,1),(2000,1,2),(2000,1,3),(2000,3,1)] :
  (int * int * int) list
val it = [(2000,1,1),(2000,1,2),(2000,1,3),(2000,3,1)] :
  (int * int * int) list
```

3.1.6 Функція get_nth:

Вхідні дані:

```
get_nth([],2);
get_nth(["One","Two","Three","Four","Five"],3);
```

Вихідні дані:

```
val get_nth = fn : string list * int -> string
val it = "" : string
val it = "Three" : string
```

3.1.7 Функція date_to_string:

Вхідні дані:

```
date_to_string((2000,1,1));
```

Вихідні дані:

```
[autoloading]
[library $SMLNJ-BASIS/basis.cm is stable]
[library $SMLNJ-BASIS/(basis.cm):basis-common.cm is stable]
[autoloading done]
val date_to_string = fn : int * int * int -> string
val it = "January 1, 2000" : string
```

3.1.8 Функція number_before_reaching_sum

Вхідні дані:

```
number_before_reaching_sum(10,[1,2,3,4,5,6]);
number_before_reaching_sum(10,[]);
```

Вихідні дані:

```
val number_before_reaching_sum = fn : int * int list -> int
val it = 3 : int
val it = 0 : int
```

3.1.9 Функція what_month:

Вхідні дані:

```
what_month(60);
```

Вихідні дані:

```
val what_month = fn : int -> int
val it = 3 : int
```

3.1.10 Функція month_range:

Вхідні дані:

```
month_range(58,61);
```

Вихідні дані:

```
val month_range = fn : int * int -> int list
val it = [2,2,3,3] : int list
```

3.1.11 Функція oldest:

Вхідні дані:

```
oldest([]);
oldest([(2000,1,1), (2000,1,2), (2000,1,3), (2000,2,1), (2000,2,2), (2000,3,1)]);
oldest([(2000,1,1)]);
```

Вихідні дані:

```
val oldest = fn : (int * int * int) list -> string
val it = "NONE" : string
val it = "SOME March 1, 2000" : string
val it = "SOME January 1, 2000" : string
```

4 ВИСНОВОК

Під час виконання даної лабораторної роботи я дослідив функціональний підхід написання програми за допомогою мови програмування - SML. Даний підхід дозволяє програмісту ефективно вирішувати поставлені завдання. Він є універсальним, використовується у вирішенні завдань сьогодення.