First steps with Caml
Be lazy BUT efficient!

Start-up

Exercise 1 (Peculiar)

```
max_int ;;
max_int + 1 ;;
min_int ;;
min_int * 2 ;;
max_int * 2 ;;

exp 40. = exp 40. +. 1. ;;     (* exp: exponential function *)
exp 40. = exp 40. +. 15. ;;
exp 40. = exp 40. +. 16. ;;
```

Evaluate these phrases. Observe and explain the results.

Exercise 2 (Functions)

Download the online file `practical1_functions.ml`. Then open it in `emacs` / `Caml` .
Evaluate these phrases, observe the results and try to understand the rules.

Exercise 3 (Exceptions)

Download the online file `practical1_exceptions.ml`.

Study these examples on the use of the functions `failwith` and `invalid_arg` that raise exceptions. From now, you have to use these functions to manage error cases.

```
val invalid_arg :  string -> 'a
```

Raise exception Invalid_argument with the given string.

```
val failwith :  string -> 'a
```

Raise exception Failure with the given string.

Riddles

Exercise 4 (Mystery number)

Let *myst* be a positive three digit integer with the following properties:

- *myst* is a perfect square[1] ;
- the sum of the squares of the two highest digits of *myst* is also a perfect square.

Write the Caml function `mystery:int->bool` testing whether a given integer is the mystery number *myst*. The function has to raise an exception if the parameter is not a three digit positive number.

```
# mystery 123 ;;
- : bool = false
# mystery 12345 ;;
Exception: Invalid_argument "mystery: "not a 3-digits integer".
```

Useful functions:

- `val sqrt : float -> float` Square root.
- `val int_of_float : float -> int` Truncate the given floating-point number to an integer.
- `val float_of_int : int -> float` Convert an integer to floating-point.

Subsidiary question: Does such a number exist?

Exercise 5 (Crack the Code)

In the school, there is a safe that contains all exam subjects. The code C is a four-digit number such that:

- the 4 digits of C are even ;
- the sum of the 4 digits of C is 8 ;
- C is divisible by 23 ;
- by adding to C its "mirror" (4321 is the mirror of 1234), we obtain a 4-digit palindromic number: an integer identical to its "mirror" (e.g. 1991).

Write a Caml function testing whether a given integer satisfies these conditions.
The function has to raise an exception if the parameter is not a four digit positive integer. *Subsidiary question:* What is the code?

Exercise 6 (The garden surface area)

Paul and Virginia consider the garden total surface area (an integer of square meters). Paul decreases this number by 5000. Then Virginia writes 2 times, side by side, the 3-digit number remaining. Then Paul divides this new number by 7, and substracts 7 times from it the remainder of the division. Then Virginia divides this new number by 11, and substracts 11 times from it the remainder of the division. To finish, Paul divides this new number by 13, and substracts 13 times from it the remainder of the division. The result is 555.

Write the Caml function `surface` that tests whether a given integer may correspond to the surface of the garden of Paul and Virginia.

```
# surface ;;
- : int -> bool = <fun>
# surface (-50) ;;
Exception: Invalid_argument "surface: negative parameter".
# surface 1234 ;;
- : bool = false
# surface ??? ;;
- : bool = true
```

[1] Perfect square = the square of an integer.

Dates

Exercise 7 (The day after)

Write a function that calculates the day after a given date. The function has to raise an exception if the date is not valid.

Level 1

- In parameter, a date is represented by 3 integers : day month year.
- As a result, a date is represented by a tuple (day, month, year).

```
# day_after 12 9 2024 ;;
- : int * int * int = (13, 9, 2024)
# day_after 28 2 2024 ;;
- : int * int * int = (29, 2, 2024)
# day_after 28 2 2025 ;;
- : int * int * int = (1, 3, 2025)
# day_after 31 12 2024 ;;
- : int * int * int = (1, 1, 2025)
# day_after 12 15 2005 ;;
- : int * int * int = (13, 15, 2005)

# day_after 12 15 2005 ;;
Exception: Failure "the day after:not a month number".
# day_after 31 6 2029 ;;
Exception: Failure "the day after:this day does not exist for month 6".
```

Level 2

The dates (parameter and result) must have the following shape "jj/mm/aaaa" (French shape).

```
    # tomorrow "31/12/2023" ;;       (* French shape *)
    - : string = "01/01/2024"
    # tomorrow "28/02/2020" ;;
    - : string = "29/02/2020"
    # tomorrow "45/02/2005" ;;
    Exception: Failure "tomorrow:not a valid day".
```

More: What if we want to process dates with this shape: "December 24, 2020"?

```
    # tomorrow "December 24, 2023" ;;
    - : string = "December 25, 2023"
```

Reminder:

```
    # let s = "Hello" ^ " " ^ "World" ;;
    val s : string = "Hello World"
    # String.length s ;;
    - : int = 11
```

Other useful functions in the module `String`

val sub : string -> int -> int -> string

`String.sub` *s start len* returns a fresh string of length *len*, containing the substring of *s* that starts at position *start* and has length *len*.

Raise `Invalid_argument` if *start* and *len* do not designate a valid substring of *s*.

val int_of_string : string -> int

Convert the given string to an integer. The string is read in decimal (by default) or in hexadecimal (if it begins with 0x or 0X), octal (if it begins with 0o or 0O), or binary (if it begins with 0b or 0B).

Raise `Failure "int_of_string"` if the given string is not a valid representation of an integer, or if the integer represented exceeds the range of integers representable in type int.