# Assignment 3
# Itemizations and structures

**Programming Fundamentals 1**

Submit by: Wednesday, 11 October 2023 at 20:30

## 1 The language

Develop this assignment in DrRacket, using the **Beginning Student Language** (BSL). The official documentation for BSL is available here (as well as through DrRacket's *Help* menu).

You must follow the design recipe to solve the following exercises. That is, you should include for every function definition:

- data type definitions, written as comments and `struct` declarations, for all the function's input or output types that are not built-in types (the built-in types are `Number`, `String`, `Boolean`, `Image`, and `Posn`)[1]
- examples of instances of every `struct` data type definition
- a signature, purpose statement, and header
- a template, commented out
- several input/output examples, in the form of executable tests

Your functions need not be "robust". That is, you can assume that the actual arguments passed to your functions agree with the function signature.

[1] If an exercise uses a type that you defined for a previous exercise in this assignment, you do not need to define it again but can just use it in the other steps of the recipe.

## 2 The assignment

1. Design a function `median` that takes three numbers and returns their *median*—that is, the number that lies in the middle once the three numbers are ordered from smallest to largest. For example, the median of $4, 2, 8$ is $4$.

2. Design a function `3-max` that takes three numbers and returns their *maximum*—that is, the largest number among them. (You *cannot* use Racket's built-in function `max` to implement `3-max`.)

### 2.1 Data type `Posn`

The following exercises use built-in structure type `Posn` to model *points* given as bidimensional coordinates.

3. Design a function `left-of?` that inputs two points, and determines if the second lies to the left of the first. *Hint*: compare the $x$ coordinates of the two points.

4. Design a function `mirror-vertical` that inputs a point a returns the point that lies at the same height as the input but at the opposite horizontal coordinate. For example the vertical mirror of $(3, 7)$ is $(-3, 7)$.

### 2.2 Dates

5. Write a data type definition for the months of the year, represented as `String`s: `"January"`, `"February"`, ...

6. Design a function `number->month` that takes a month number (an integer from 1 to 12) and converts it to the month's name. For example, it should output `"March"` when the input is 3.

7. Design a function `leap-year?` that takes a year (an integer) and determines whether it is a leap year.

   A year is a leap year if it is an integer multiple of 4 but not of 100 (unless it is also an integer multiple of 400). For example: 1868 is a leap year (multiple of 4); 1900 is not (multiple of 4 but also of 100 and not of 400); 1904 is a leap year (multiple of 4 and not of 100) and so is 2000 (multiple of 4, 100, and 400). Use function `modulo` to determine if a number is an integer multiple of another one.

8. Write a data type definition for a `Date` of the year, consisting of: a day (an integer between 1 and 31), a month (an integer between 1 and 12), and a year (any integer except 0, which is not a valid year).

9. Design a function `date->string` that takes a `Date` and returns a `String` with a representation of the date of the form `"[Day] [Month] [Year]"`. The day in the output should be a number; the month should be a name; the year should be a positive integer, with `"BCE"` appended if the year is negative. For example, date $30\ \ 3\ \ {-1204}$ should render as `"30 March 1204 BCE"`; date $14\ \ 9\ \ 2020$ should render as `"14 September 2020"`. Use function `number->month` defined above in the definition of `date->string`.

10. Design a function `days-in-month` that takes a month number (an integer from 1 to 12) and returns the number of days in that month in a non-leap year. Remember that there are 30 days in November, April, and June, and September; 28 days in February (in a non-leap year); and 31 days in all other months.

11. Design a function `date-valid?` that takes a `Date` and determines if the date is valid, that is if the month number is between 1 and 12; the day number is consistent with the month; and the year number is not zero. Remember that 29 is a valid day number in February only in leap years. Use any functions defined above in the implementation of `date-valid?`.

## 3 How and what to turn in

Using *iCorsi*'s website for Programming Fundamentals 1, upload under **Assignment 3** a single Racket source file named `YourLastName_YourFirstName_PF1_Assignment3.rkt` including *all function definitions and data type declarations* described in this assignment, each with the *design recipe*'s artifacts.

The submission **deadline** is hard, and late submissions will not be accepted. If you have a justified reason that prevents you from submitting this assignment on time, ask the instructors for an extension **well before** the deadline.

## 4 Points

This assignment awards a maximum of 1 point points (out of the 100 points awarded in the whole course).

## 5 Plagiarism policy

Assignments must be done alone. Students are allowed to generally discuss assignments and solutions among them, but each student must work on and write down their assignment independent of other students. In particular, both sharing solutions of assignments among students and reusing solutions of assignments done by students of previous years or by anyone else are not allowed and constitute cheating.

The same policy applies to any material and examples that may be available online: you are allowed to look them up, but you are required to write down your solution completely on your own without directly reusing others' work. Failure to do so will be considered plagiarism.

## ChatGPT, CoPilot & Co.

The plagiarism policy also applies to AI-based tools such as ChatGPT or CoPilot:

1. We recommend that you try to work on each problem on your own, especially when you are still unfamiliar with the techniques and language that is used. This is the most productive approach pedagogically, as it helps make you fully understand and learn – and thus makes for the fastest *learning* progress.

2. If you find it useful – for example to check your solution before submitting it, or to give you a hint if you get "stuck" – you may seek the help of AI-based tools such as ChatGPT or CoPilot. However, you remain entirely responsible for the solution that you submit: its correctness, quality, and accuracy.

3. Be aware that AI-based tools may be unreliable, and provide no guarantee of correctly interpreting your prompts. They can provide useful help, but only if you have enough experience and knowledge to be able to assess the quality of their output.

4. If you use any AI-based tools to work on this assignment, you must declare it in the solution you submit. At the bottom of the Racket file you submit, add a paragraph of text (marked as comments), mentioning which tool(s) you used, how you used it, and for what tasks.

Failure to abide by these rules, including failing to disclose the usage of AI-based tools, will be considered plagiarism.

Remember that cheating and plagiarism are unacceptable. The penalty for cheating or copying – including allowing others to copy your work – is up to 100% of your grade for the course.