# LANGUAGES AND ALGORITHM FOR ARTIFICIAL INTELLIGENCE PROJECT

A.A. 2021/2022

First Module

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

TEAM

Samuele Marino

Vincenzo Collura

Alice Gibellato

Prof. Maurizio Gabbrielli

17/02/2022

# INTRODUCTION

The project chosen by the group aims to solve the exercises of the "Allenamenti giochi d'autunno 2021" issued by Bocconi University. The group chose to carry out all the exercises using MinZink. Subsequently, the progress and considerations made during the implementation will be explained.

# 1. ROSSO E NERO

To represent the problem we are considering the number of cards. The red cards in the first deck and the total number of them were declared as parameters of type int, unlike the number of black cards in the first and second deck and the number of red cards in the second deck are declared as variables with a range between 0 and 26.

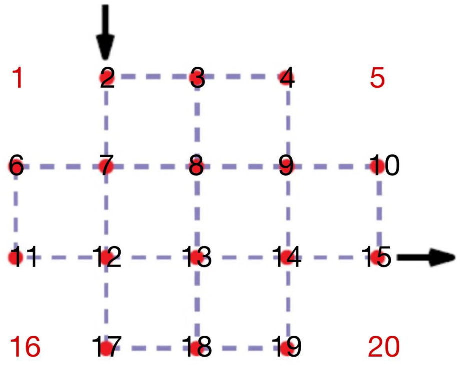
The constraints, which we have reported below, represent all the conditions that must be satisfied:

- The number of black cards in the first deck is equal to the total number of cards (25) minus the number of red cards in the deck.
- The number of red cards in the second deck is equal to the number of total red cards minus the number of red cards in the first deck.
- The number of black cards in the second deck is equal to the number of total cards in the second deck (27) minus its red cards.

The program computes the number of black cards in the first deck and the number of red and black cards in the second deck. In conclusion we can say that the number of black cards in the second deck is equal to 13.

# 2. IL LABIRINTO

A 4x5 matrix is used to represent the labyrinth as follow:



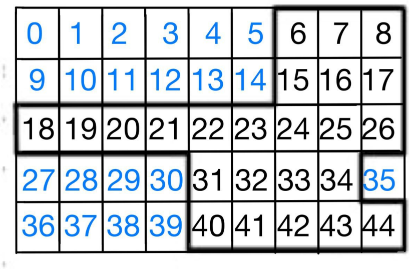
An array of 16 elements has been initialized in a range of 2 to 19. It represents the path taken to exit the labyrinth and when it reaches its final position the array will always be completed with 15 so that it can never go back. The variable *distance* representing the length of the path is initialized.

The constraints, which we have reported below, represent all the conditions that must be satisfied:

- The first value of the array must be equal to 2 (labyrinth start point).
- If you get to the end (15), all the following elements of the array are set to 15.
- Imposes the possible route.
- Points that cannot be reached are declared.
- You can reach each point of the labyrinth just once
- It requires leaving the labyrinth.

The programme maximizes the distance that can be traveled which is 150.

# 3. IL CARTONE



|  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 |
| 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 |

In order to represent the cardboard we use a matrix of zeros and ones. The ones represent the real numbers of the cardboard because we are inscribing it inside the matrix. Then two arrays are used to represent start and end indices of all possible rectangle/square splitting. These arrays are initialized in a range from 0 to 23. To represent the final splitting we use an array of range from 0 to 44.

The cardboard is divided into rectangles and squares where the points of the array that belong to the same figure are represented by the same number.
The constraints, which we have reported below, represent all the conditions that must be satisfied:

- Two constraints have been initialized using *farall()* to represent the start and end indexes.
- A constraint is used to impose that the figures found must be rectangles or squares.
- A constraint has been implemented so that the found figures are not overlapped.

The solution, which is equal to 4, is found by minimizing the maximum number found within the matrix. This number represents the number of figures that the cardboard has been divided into.

## 4. IL PIU' PICCOLO

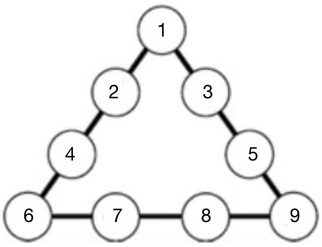
An array of 4 elements with a range of values from 0 to 9 has been declared to represent the number.
The constraints, which we have reported below, represent all the conditions that must be satisfied:

- In order to have all digits different, the *alldifferent()* function was called on the values of the array.
- A number is multiple 11 if and only if the absolute value of the difference between the sum of the even-numbered digits and the sum of the odd-numbered digits is equal to 0.
- The first value of the array has been set greater than 0 because the number cannot start with the digit 0.
- In order to have all digits even, a *forall()* has been used which imposes the rest of the division by two equal to zero for each digit.

A *number()* function is declared which converts the array values into a number using multiples of 10.
The programme finally minimizes the value of the *number()* function and returns the smallest number found, which is 2046.

## 5. UN TRIANGOLO MAGICO

An array of 9 values in a range from 1 to 9 representing the triangle has been initialized as in the picture.



The constraints, which we have reported below, represent all the conditions that must be satisfied:

- With *alldifferent()* every value around the triangle is different.
- The values in the triangle already imposed by the problem have been declared.
- Three constraints have been initialized which require the sum of the numbers on each side to be equal to 20.

The problem asks which number is in the lower left-hand corner and the program shows the correct answer which is 9.

## 6. UN TRIANGOLO MASSIMO

Three variables representing the sums of the 3 sides have been initialized with a range of values from 1 to 30.
An array of 9 values representing the numbers to be placed at the sides of the triangle has been initialized.
The constraints, which we have reported below, represent all the conditions that must be satisfied:

- With *alldifferent()* every value around the triangle is different.
- Three constraints representing the sum of the 3 sides have been initialized.

The *totalsum()* function represents the sum of the three sides.
The solution of the problem maximizes the sum of the sides and shows the correct answer to be 69.

## 7. QUANTI 9!

A starting variable has been initialized with a value between 10 and 99.
A constraint has been initialized using *forall()* that multiplies the number by its first 11 multiples and ensures that at least one of these conditions is satisfied:

- Division by 100 of a multiple is equal to 9.
- The remainder of the division by 100 of a multiple divided by 10 is equal to 9.
- The remainder of the division by 10 of a multiple is equal to 9.

The problem computes the starting number which is equal to 99.

## 8. L’ETA’ DI MATTEO

An array of 4 elements with a range of values from 0 to 9 was declared to represent the year and a variable has also been declared to represent the year of birth. This last variable is initialized with a range from 2000 to 2150 (indicative of a person's age) and consists of the array values multiplied by multiples of 10 to represent the year.
A variable representing the sum of the digits in the array has been declared and this variable is between 0 and 36 because the sum of the digits can be at most 36.
A variable representing age equal to the year minus 2000 was initialized.
The constraints that have been declared are equal to age divided by 5.
The program computes the years representing ⅕ of Matteo’s age equal to 2020 and 2065. To display all correct values the compiler must be set to *--all-solutions.*

## 9. QUATTRO CIFRE PER UNA DATA

An array of 8 digits in a range from 0 to 9 representing the digits of the date has been implemented. Three variables have been implemented as day, month and year and represent the values of the array multiplied by multiples of 10.
The constraints, which we have reported below, represent all the conditions that must be satisfied:

- A *forall()* is used to check that the digits within the array are contained twice.
- Two constraints have been initialized to ensure that the date found is greater than 19.09.2021.

The *recentdate()* function sums the values of days, months and years multiplied by multiples of 10.
The solution to the problem minimizes the *recentdate()* function and finds the first subsequent date written with 4 digits that is equal to 13.01.2023.

## 10. E ADESSO SONO 18!

A value of type int represents the number of white squares (18). Four variables have been declared with a range from 3 to 9 representing the number of white squares at the top, bottom, right and left respectively. The white squares placed around the figure together with the gray squares in the center must represent a rectangle.
The constraints, which we have reported below, represent all the conditions that must be satisfied:

- The number of white squares at the top must be equal to the number of white squares at the bottom.
- The number of white squares on the right must be equal to the number of white squares on the left.
- The sum of the top, bottom, right and left white squares minus 4 (perimeter of the rectangle) represents the number of white squares.

The number of gray squares represents the number of white squares at the top minus two multiplied by the number of squares on the right minus two.
The result of the program shows the number of gray squares that is 12. The function is maximized because the aim is to find the maximum number.

## 11. IL NONO

An array of 4 values in a range from 0 to 9 representing the digits of the number has been declared. An array of 3 values in a range from 0 to 9 has been declared, which represent the digits of the ninth number represented.

Two variables representing the starting number and the ninth number have been declared. These variables are the multiplication of the array values times multiples of 10.

The constraints, which we have reported below, represent all the conditions that must be satisfied:

- It is imposed that there be a zero in the second, third or fourth digit.
- Three constraints have been declared which, depending on the position of zero, move the array values to the left.
- The value of the final number must be equal to the starting number divided by 9.

The value of the final number must be minimized because the problem asks for the smallest number. The result of the program equals the correct answer and it is 2025.

## 12. MAGIA!

A variable representing the starting number and a variable representing the final number.

The constraints, which we have reported below, represent all the conditions that must be satisfied:

- The final number must be equal to the starting number multiplied by 4 minus 3
- The remainder of the final number divided by 10 must be equal to the initial number divided by 10.
- The final number divided by 10 must be equal to the remainder of the division by 10 of the initial number

The problem computes the initial number which is equal to 16

## 13. LE SCATOLE DI CARLA

The values of the sides of the two cubes were initialized and a variable *x* has been declared to represent the number of times the large cube is filled using the small one.

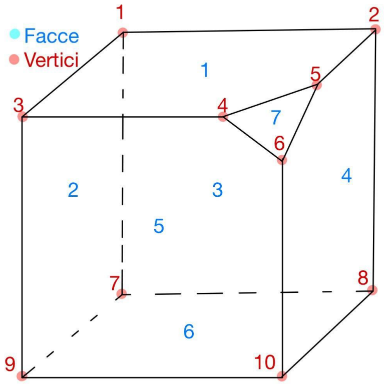
The constraint indicates that the area of the small square multiplied by x must be equal to the area of the large square.

The problem computes the number of times the small cube is used to fill the large one, which equals 8.

## 14. IL SOLIDO DI LEGNO

A variable *c* representing the number of competing faces and edges in the same vertex has been implemented.

The total number of faces was defined (7) and two variables representing the total number of edges and vertices have been implemented.



The constraints, which we have reported below, represent all the conditions that must be satisfied:

- The total number of edges is equal to the number of faces plus the number of vertices minus 2.
- The number of vertices is equal to twice the number of edges divided by c.

If a vertex is cut from a solid figure, the number of faces increases by one, therefore the final number of faces is equal to the number of vertices plus the number of faces.

The solution to the problem computes the maximum number of faces obtained, which is 17.

## 15. RENATO FA IL FURBO

The number of square meters is defined (2021) and the variable total number of square meters was initialized in a range from 100 to 2021.

Four variables have been defined representing the top, bottom, right and left sides respectively.

The constraints, which we have reported below, represent all the conditions that must be satisfied:

- Total number of square meters is equal to square meters minus 5 for the right-hand side.
- The top side must be greater than the right side.
- The top side must be the same as the bottom side.
- The right side must be the same as the left side.
- The top side multiplied by the right side equals the number of square meters.

The solution to the problem computes the total square meters that Renato actually has to cut, which equals 1806.

## 16. LA SVEGLIA DI LUCA

An array *time* of 6 values in the range 0 to 5 representing the numbers of the alarm clock has been initialized.

Three variables representing hours, minutes and seconds have been declared as multiplication of the array values times multiples of 10. The constraint requires that the elements of time are all different.

The program finds 312 solutions and shows how many times the alarm clock indicates a 0, a 1, a 2, a 3, a 4 and a 5. To display all correct values the compiler must be set to *--all-solutions --statistics.*

## 17. CON LE DIECI CIFRE

A variable *x* representing the initial number has been initialized. An array of 16 values has been initialized, containing the digits of the initial number raised to the third in the first 9 positions and the digits of the same number raised to the fourth in the following positions.

The constraints, which we have reported below, represent all the conditions that must be satisfied:

- Two constraints representing n3 as the number raised to the third and n4 as the number raised to the fourth respectively.
- Constraint using a *forall()* so that all digits from 0 to 9 are used.

The program results in 18, which is the value of the initial number.

## 18. UN TRIANGOLO NELL'OROLOGIO

In order to represent a traditional clock movement we divide it into 60 equal parts. The second hand can move freely, while the minute hand changes position with each complete revolution of the second hand. For the hour hand movement we divide the clock in 12 parts. In each of these parts the hour hand can move 60 times one step for every minute.

An array representing the value of hours, minutes and seconds has been initialized. For the hours, 720 (60x12) parts represent all possible hand movements. For the minutes, the range of values goes from 0 to 15 because the control is carried out only on a quarter of the clock. This choice has been made because, since the movement is cyclical, if there is no combination that satisfies the constraints in the first quarter then there is no combination in the others. For the seconds, a value from 0 to 59 was considered.

The constraints, which we have reported below, represent all the conditions that must be satisfied:

- Hours must be equal to minutes times 60 to represent real clock movement.
- Minutes minus the hours divided by sixty must be equal to 20 because the hands should be twenty minutes apart.
- Seconds minus minutes must be equal to 20.

The constraints are unsatisfied because there is no case in which the hands shape an equilateral triangle inside the watch.