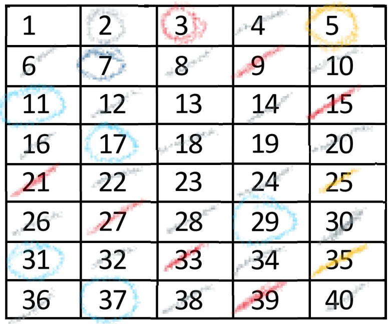
# Practical Work N°1: Linked List

- You have one month to submit your work on Classroom (from 4 March to 8 April).
- This work must be done by two students.
- You must use Code::Blocks and submit a compressed project entitled: LName1_FName1_LName2_FName2_Class_PW1.
- Work that does not use the abstract machine will not be examined.
- Each question must be resolved, organized and commented in a separate file and consist of a library that can be included and used. Otherwise, sanctions are applied.

We aim to obtain all prime numbers less than or equal to a specified number *n*. We can use either a dynamic array or a list to represent the numbers. Our primary goal is to analyse and compare the performance of these two data structures in terms of iteration count and, consequently, their execution time. Additionally, we look to optimize the search process by exploring the structure and removing all multiples instead of checking the divisors of each number using the following schema:



| | | | | |
|---|---|---|---|---|
| 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~~ |

As you can see, we can start by generating only odd numbers between two and the given number n. Then we gradually remove the multiples of each number. The remaining ones are the prime numbers.

## Questions :

1. Implement the search algorithm without using the structures.
2. Implement the two data structures and solve the problem using the method described in the table above.
3. Index the list by range to allow printing of prime numbers between two values.
4. Use the list of prime numbers to express each number of the first list as the product of prime numbers. Next, you add the found divisors to the node.
5. Find the co-prime numbers
6. Measure the complexity and number of iterations.