# Banker's Algorithm

This project was written in C# and consists of three Windows Forms.

## Form1

Form1 contains a single function called `Enter\_Click`. This function is responsible for taking two inputs from the user (the number of processes and the number of resources) and then closing Form1 and opening Form2.

Sure, here's the text converted to Markdown format:

## Form2

In this form, the user is able to fill out four tables: `Total Resources`, `Available Resources`, `Current Allocation`, and `Max Need`. The system then calculates the `Remaining Need` table and determines if the system is safe or not. If it is safe, the system displays the safe sequence of processes. The user also has the ability to assume if any process requests any number of resources of any resource and the system responds if it can or not.

### Functions

Form2 contains several functions:

- `Total\_ressources()`: Builds the `Total Resources` table according to the number of resources and processes that the user entered in Form1.

- `Available\_Ressources()`: Builds the `Available Resources` table. Same as `Total\_ressources()` function.

- `Current\_allocated()`: Builds the `Current Allocation` table. Same as `Total\_ressources()` function.

- `Max\_need()`: Builds the `Max Need` table. Same as `Total\_ressources()` function.

- `remain\_need()`: Calculates the `Remaining Need` table by subtracting each cell from `Max Need` table to `Current Allocation` table. Same as `Total\_ressources()` function but with an additional calculation.

- `combobo\_list()`: Fills two comboboxes. The first one consists of process names (P1, P2, ...), and the second combobox consists of resource names (R1, R2, ...).

- `common()`: Returns a boolean. It returns `true` if there isn't an empty cell in any of the tables and if each cell in `Max Need` table is greater than or equal to the corresponding cell in `Current Allocation` table, else it returns `false`.

- `enter\_button\_Click()`: Optional button that the user clicks when they make an assumption. Calls `common()` function and if it returns `true`, it checks if the user's assumption is less than or greater than the available resource.

- `Finish\_button\_Click()`: Button that calls `remain\_need()` and `logic()` functions.

- `Total\_Ressources\_Constraints()`: Checks that all the cells in `Total Resources` table are filled and that they are integer values.

- `Available\_ressources\_Constraints()`: Checks that all the cells in `Available Resources` table are filled and that they are integer values.

- `Current\_allocated\_Constraints()`: Checks that all the cells in `Current Allocation` table are filled and that they are integer values.

- `Max\_need\_constraints()`: Checks that all the cells in `Max Need` table are filled and that they are integer values.

- `logic()`: Function that implements the banker algorithm logic. It displays two labels. The first one shows if the system is safe or unsafe, and the second label shows the safe sequence of processes. It also transfers the steps in a list called `stepslist`.

- `printavailable()`: Function that returns the number of the available resources. This function is called in the `logic()` algorithm.

- `steps\_button\_Click()`: Button that hides this form and opens Form3.

## Form3

This form consists of a label and a richtextbox that contains the steps of the algorithm.

Function:

1. `Print\_steps()`: This function prints all the steps of the algorithm according to the `steps\_list