Programming 2 - Laboratories: Task 8

In this task you have to implement two classes.

Class `Digital_signal` represents digital signal of any length. This class possesses dynamically allocated array `samples`, holding signal samples. Number of samples is represented by `no_of_samples`.

Class `Signal_plot` represents simple graphical plot of digital signal. Plot is represented as a two dimensional, dynamically allocated array of chars. Size of the plot (number of rows and columns, stored in fields `no_of_rows` and `no_of_columns` respectively) is determined by number of signal samples and signal amplitude. This class possesses internally digital signal (object of `Digital_signal` class).

Part 1 (2 points)

a) Implement class `Digital_signal` constructor and destructor:

```
Digital_signal(int no_of_samples = 0, int* samples = nullptr);
~Digital_signal();
```

creating/destroying a single `Digital_signal` object. Signal might be empty. If samples values are not passed to the constructor (while a number of samples are properly defined), all signal values should be set to zero.

b) Overload operator <<

```
friend ostream& operator<<(ostream&, const Digital_signal&);
```

printing to the screen signal samples values and signal info (see example output). Additionally you need to implement methods `int min_val() const` and `int max_val() const` returning signal's minimum and maximum value respectively.

Part 2 (2 point)

Overload operator +

```
friend Digital_signal operator+(const Digital_signal&, int c);
```

which adds a constant value to each signal sample. Add to class all necessary elements!!! (see also usage in main).

Part 3 (2 points)

Make class `Signal_plot` a friend to class `Digital_signal.`

a) Implement class `Signal_plot` constructor and destructor:

```
Signal_plot(int no_of_samples = 0, int* samples = nullptr);
~Signal_plot();
```

creating/destroying a single `Signal_plot` object. Signal and plot might be empty. If samples values are not passed to the constructor (while a number of samples are properly defined), all signal values should be set to zero.

Additionally you need to implement private methods:

- `void init_plot();`
- `void update_plot();`
- `void clean_plot();`

Method `void init_plot()` should set proper values of `no_of_rows` and `no_of_columns` (if samples in signal exists) and allocate two dimensional array `plot`.

Method `void update_plot()` should set proper values in `plot` (if samples in signal exists, and plot was properly allocated). Each element of the plot connected with valid signal sample, should be set to '*'. All other elements should be set to ' ' (empty space).

Method `void clean_plot()` should free the memory and set all member fields to initial values (zeros for numbers and `nullptr` for pointer fields).

Above three methods should be used in implementation of constructors, destructor and other important class methods (if needed).

Overload operator <<

```
friend ostream& operator<<(ostream&, const Signal_plot &);
```

is given. In the task, upper left corner of the plot is indexed with [0,0].

Part 4 (2 points)

Implement:

```
Signal_plot(const Digital_signal&);
Signal_plot(const Signal_plot&);
Signal_plot& operator=(const Signal_plot&);
```

In implementation use also private methods defined in part 3.