ЛАБОРАТОРНА РОБОТА №3.

***Реалізація класу документу з використанням спеціальної***

***мови розмітки***

**1. Завдання на роботу:**

Реалізувати клас документу із (як мінімум) наступним інтерфейсом:

class Document

{

public:

Document();

Document(const Document&);

Document(Document&&) noexcept;

explicit Document(std::istream& stream);

explicit Document(const std::string& text);

Document& operator=(const Document&);

Document& operator=(Document&&) noexcept;

bool empty() const;

size\_t count() const;

void remove(size\_t idx);

void update(size\_t idx, std::istream& stream);

void update(size\_t idx, const std::string& text);

void clear();

void load(std::istream& stream);

void load(const std::string& text);

std::string render() const;

};

Клас представляє деякий документ, який дозволяє виводити форматований спеціальним чином текст на екран консолі (в текстовому режимі).

Для розмітки документа використовується спеціальна мова розмітки, яка буде описана нижче.

* 1. **Опис методів:**

1. Document() – конструктор за замовченням, створює порожній документ.
2. Document(const Document&) – конструктор копіювання, створює копію документа. При цьому, обидві копії мають надалі модифікуватися незалежно (зміни в одному документі мають не дублюватися в іншому).
3. Document(Document&&) – конструктор переміщення, виконує швидке переміщення вмісту одного документа в інший.
4. Document(std::istream& stream) – виконує завантаження (читання) документу відповідно до мови розмітки з потоку вводу.
5. Document(const std::string& text) – виконує завантаження (читання) документу відповідно до мови розмітки з рядка.
6. operator=(const Document&) – виконує копіювання вмісту документу.
7. operator=(Document&&) – виконує швидке переміщення вмісту одного документа в інший.
8. empty() – перевіряє, чи є документ порожнім.
9. count() – виводить загальну кількість елементів документу.
10. remove(size\_t idx) – видаляє елемент документу із відповідним індексом (а також всі вкладені елементи, якщо такі є). Після цього нумерація (індекси) наступних елементів може бути змінена.
11. update(size\_t idx, std::istream& stream) – перезавантажує вміст відповідного (за індексом) елемента з потоку вводу відповідно до мови розмітки (при цьому, поточний вміст елемента, включаючи вкладені елементи, видаляється). Після цього нумерація (індекси) наступних елементів може бути змінена.
12. update(size\_t idx, const std::string& text) – перезавантажує вміст відповідного (за індексом) елемента з рядка відповідно до мови розмітки (при цьому, поточний вміст елемента, включаючи вкладені елементи, видаляється). Після цього нумерація (індекси) наступних елементів може бути змінена.
13. clear() – повністю очищує вміст документа.
14. load(std::istream& stream) – завантажує документ з потоку вводу відповідно до мови розмітки. Поточний вміст документа очищується.
15. load(const std::string& text) – завантажує документ з рядка відповідно до мови розмітки. Поточний вміст документа очищується.
16. render() – виводить вміст документа в рядок.
    1. **Опис мови розмітки:**

Документ має ієрархічну структуру та складається з елементів, які бувають двох типів: ***блок*** та ***рядок***. При цьому, є один кореневий елемент.

Для елементів типу ***блок*** можливо задавати додаткові атрибути, що дозволяють форматувати його вміст.

Синтаксис елементу типу ***рядок***:

**$***рядок*

тобто, рядок вхідного тексту класифікується як елемент типу рядок, якщо починається символом $. При цьому, вміст (текст) елементу – вся інша частина рядка, включаючи пробіли (починаючи з наступного символу до кінця рядка).

Синтаксис елементу типу ***блок***:

{

#*Атрибут1*

#*Атрибут2*

*Вміст*

}

тобто, елемент типу ***блок*** розміщується в фігурних дужках, може містити деяку кількість рядків-атрибутів, що починаються символом #, та містить деякий вміст, що може бути представлений довільною кількістю інших елементів (***рядків*** чи ***блоків***).

Таким чином, елемент типу рядок дозволяє визначити вміст (текст) документа, а елемент ***блок*** дозволяє використовувати одночасно декілька елементів, групувати їх та форматувати за допомогою атрибутів.

Вкладені елементи розміщуються в ***блоках*** один під іншим (тобто, елементи не групуються по горизонталі). При цьому, ширина ***блоку*** визначається автоматично відповідно до його вмісту. Елементи-***рядки*** розміщуються в одному рядку (перенос слів не виконується).

Рядок атрибуту мають наступний синтаксис:

#*атрибут параметр1 параметр2*

тобто, атрибут – це рядок, що починається з символу #, містить ім’я атрибута та можливі параметри (відповідно до атрибуту).

Необхідно реалізувати як мінімум наступні атрибути:

1. align *тип* – виконує вирівнювання вмісту блоку відповідно до значення параметру *тип*:
   1. left – вирівнювання по лівому краю.
   2. center – вирівнювання по центру (якщо рівно по центру розмістити неможливо, то більшим має бути лівий відступ).
   3. right – вирівнювання по правому краю.

За замовченням вирівнювання – по лівому краю, але вкладені блоки займають всю ширину, на відміну від конкретних режимів вирівнювання, в яких блок займає лише необхідний розмір.

1. border *розмір* *символ* – задає рамку навколо блоку, розміром «*розмір*» символів, при цьому рамка малюється символом «*символ*»; якщо символ не вказано, то використовується символ «пробіл». За замовченням рамка не малюється.
2. margin *top right bottom left* – задає відступи навколо блоку (ззовні рамки, навіть якщо її немає) з кожного боку відповідно до значень параметрів. За замовченням відступи відсутні.
3. padding *top right bottom left* – задає відступи до вмісту всередині блоку (всередині рамки, навіть якщо її немає) з кожного боку відповідно до значень параметрів. За замовченням відступи відсутні.
4. mask *символ* – виконує заміщення всіх непробільних символів у вмісті блоку (не включаючи його рамку, але включаючи рамки вкладених блоків) заданим символом (якщо символ не задано, то використовується «пробіл»).
5. width *значення* – встановлює ширину вмісту блоку не меншою за вказане значення (навіть якщо реальний вміст має менший розмір).

Всі інші рядки в тексті розмітки не мають значення та можуть бути проігноровані (дозволяється та вітається розширення мови розмітки додатковими командами, але так, щоб вони не «заважали» обов’язковому функціоналу, тобто, щоб при їх невикористанні поведінка відповідала описаній вище).

* 1. **Приклад:**

Текст розмітки:

{

#align center

#border 1 %

#margin 1 1 1 1

#padding 2 2 2 2

$a b c d e f g

{

#border 3 \*

#padding 3 3 3 3

#align left

$1234567890

$ abcdefghijklmnopqrstuvwxyz

{

#mask $

#align right

{

#margin 2 2 2 2

#padding 1 1 1 1

#border 2 +

#width 8

$ASDF

}

$ZxCvBnM

}

}

$QWERTYUIOP

{

$ ABCDEFGHIJKLMNOPQRSTUVWXYZ

{

#margin 2 3 2 3

#padding 2 1 2 1

#border 3 %

#align center

$ 12345

}

}

}

Результат виводу (render):

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %

% %

% a b c d e f g %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* 1234567890 \*\*\* %

% \*\*\* abcdefghijklmnopqrstuvwxyz \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* $$ $$ \*\*\* %

% \*\*\* $$ $$$$ $$ \*\*\* %

% \*\*\* $$ $$ \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* $$$$$$$ \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% QWERTYUIOP %

% ABCDEFGHIJKLMNOPQRSTUVWXYZ %

% %

% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%% %%% %

% %%% %%% %

% %%% 12345 %%% %

% %%% %%% %

% %%% %%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %

% %

% %

% %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Нумерація елементів (для методів remove та update) проводиться відповідно до їх появи в тексті розмітки, тобто, в наведеному прикладі елементи матимуть наступні індекси:

{ *index=0*

#align center

#border 1 %

#margin 1 1 1 1

#padding 2 2 2 2

$a b c d e f g *index=1*

{ *index=2*

#border 3 \*

#padding 3 3 3 3

#align left

$1234567890 *index=3*

$ abcdefghijklmnopqrstuvwxyz *index=4*

{ *index=5*

#mask $

#align right

{ *index=6*

#margin 2 2 2 2

#padding 1 1 1 1

#border 2 +

#width 8

$ASDF *index=7*

}

$ZxCvBnM *index=8*

}

}

$QWERTYUIOP *index=9*

{ *index=10*

$ ABCDEFGHIJKLMNOPQRSTUVWXYZ *index=11*

{ *index=12*

#margin 2 3 2 3

#padding 2 1 2 1

#border 3 %

#align center

$ 12345 *index=13*

}

}

}

Таким чином, кореневий елемент завжди має індекс 0. Його видалення (remove) призводить до очищення документа (рівносильна clear), а заміна (update) – до оновлення всього документа (рівносильна load).

Наприклад, після виконання виклику *update* для блоку з індексом 3 із наступним текстом:

{

#align center

#border 1 %

#padding 2 5 2 5

$Hello!

}

документ набуде наступного вигляду (в результаті виклику render):

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %

% %

% a b c d e f g %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* %%%%%%%%%%%%%%%%%% \*\*\* %

% \*\*\* % % \*\*\* %

% \*\*\* % % \*\*\* %

% \*\*\* % Hello! % \*\*\* %

% \*\*\* % % \*\*\* %

% \*\*\* % % \*\*\* %

% \*\*\* %%%%%%%%%%%%%%%%%% \*\*\* %

% \*\*\* abcdefghijklmnopqrstuvwxyz \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* $$ $$ \*\*\* %

% \*\*\* $$ $$$$ $$ \*\*\* %

% \*\*\* $$ $$ \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* $$$$$$$$$$$$$$ \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* $$$$$$$ \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\* \*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %

% QWERTYUIOP %

% ABCDEFGHIJKLMNOPQRSTUVWXYZ %

% %

% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%% %%% %

% %%% %%% %

% %%% 12345 %%% %

% %%% %%% %

% %%% %%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %%%%%%%%%%%%%%%%%%%%%%%% %

% %

% %

% %

% %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%