3 MarkUp 文本编辑器

需要提交的文件：*markup.h*。

你的任务是完成类 `MarkUp`，这是一个每个元素是一个 `char` 的链表的链表实现的文本编辑器。

这个文本编辑器可以被理解为二维不定长度数组，但是我们用链表套链表来实现。具体来说，每一行是一个把这一行所有字符串起来的一个链表，然后我们把每一行的链表头连起来，又成为一个链表。其中每个 `node` 我们储存一个不为 `\n` 的 `char`。具体结构可为：

```
headNode_headPointer
     |(nextHead)
headNode_row_1 -> (rowHead) Node('A') -> Node('C') -> Node('M') -> ...
     |(nextHead)
headNode_row_2 -> (rowHead) Node('2') -> Node('2') -> ...
     |(nextHead)
    ...
```

例如以上结构所指代的文本为：

> ACM
> 22

本题应该实现的函数包括：

#	函数	解释
-1	`print()`	先输出“==========MARK_UP_OUTPUT==========\n”，然后打印文本
0	constructor, destructor	构造函数、析构函数
1	`typeIn(ch)`	在全文最末尾打入字符 `ch`
2	`insert(row, col, ch)`	在第 `row` 行, 第 `col` 个位置插入字符 `ch`
3	`erase(row, col)`	删除第 `row` 行, 第 `col` 个字符
4	`query(row, col)`	返回第 `row` 行, 第 `col` 个字符
5	`split(row, col)`	相当于在 `row` 行, 第 `col` 个空位置插入换行符，该行裂成两行
6	`copy_paste(a1, a2, b1, b2, c1, c2)`	将从【a1 行第 a2 个字符】到【b1 行第 b2 个字符】这段内容复制并插入到【c1 行第 c2 个位置】处

具体解释：

以下列文本为例：

> ABCDEFG
> HIJKLMN

`print`：

需要注意，文末的空行也要输出来。

`query`，`erase`：

一个字所在的位置我们用第几行第几个字的坐标来表示。所有的位置都是 1-based 的，例如：在上述文本中，

'A' 的位置是第一行第一个字（1，1），'I' 的位置是第二行第二个字（2，2）。

`insert`：

如果我们对其进行插入，则其插入的位置也都是 1-based 的。例如，在上述文本中在 'G' 字后面加 ','，插入的位置为（1，8），对应的函数调用为：`insert(1, 8, ',')`。测试中不会插入到新行，例如对以上文本执行 `insert(3, 1, 'O')` 的操作是不会出现的，`copy_paste` 的 paste 部分也满足该要求。

`typeIn`：

直接插入在全文最末。

`split`：

换行。例如对上述文本进行 `split(1, 3)`，将会得到：

> AB
> CDEFG
> HIJKLMN

- 若换行发生在边界，则会产生空行。例如，对上述文本进行 `split(1, 8)` 将会得到：

> ABCDEFG
> //（这是一个空行）
> HIJKLMN

- `split(1, 1)` 将得到：

> //（这是一个空行）
> ABCDEFG
> HIJKLMN

`copy_paste`:

需要注意复制的内容是包含两端的字的。

另外插入（粘贴）的规则与 `insert` 相同。为简单起见，测试点中的复制区域与粘贴区域**不会**重叠。

额外说明

1. 若不使用链表套链表结构，而是使用单层链表等不符合题目要求的实现方法通过本题，你在本题中将获得不超过 80% 的分数。
2. 保证所有测试点所有输入合法或无输入。

测试点说明

测试点	测试的函数	内存泄漏检查	分数
1	-1, 0, 1	无	6
2	-1, 0, 1	无	6
3	-1, 0, 1	无	6
4	-1, 0, 1, 2, 3, 4	无	6
5	-1, 0, 1, 2, 3, 4	无	6
6	-1, 0, 1, 2, 3, 4	无	6
7	-1, 0, 1, 2, 3, 4, 5	无	6
8	-1, 0, 1, 2, 3, 4, 5	无	6
9	-1, 0, 1, 2, 3, 4, 5	无	6
10	-1, 0, 1, 2, 3, 4, 5	无	6
11	All	无	6
12	All	无	6
13	All	无	6
14	All	无	6
15	All	无	6
16–20	同 11–15	有	10

样例程序（sample.cpp）

```cpp
// correctness: typeIn, insert, erase, split, query, copy_paste

#include <iostream>
#include "markup.h"

using namespace sjtu;

MarkUp mu;

int main() {
  mu.typeIn('A');
  mu.typeIn('S');
  mu.typeIn('O');
  mu.typeIn('U');
  mu.typeIn('L');

  for (int i = 1; i <= 5; i++) {
    std::cout << mu.query(1, i) << ' ';
  }
```

```
    std::cout << std::endl;
    // A S O U L


    mu.insert(1, 2, '-');
    mu.print();
    // A-SOUL

    mu.split(1, 7);
    mu.typeIn('1');
    mu.typeIn('-');
    mu.copy_paste(1, 1, 1, 6, 2, 3);

    mu.print();
    // A-SOUL
    // 1-A-SOUL

    mu.erase(2, 2);


    mu.print();
    // A-SOUL
    // 1A-SOUL

    mu.split(2, 1);
    mu.print();
    // A-SOUL
    //
    // 1A-SOUL

    return 0;
}
```

样例输出（sample.ans）

```
A S O U L
==========MARK_UP_OUTPUT==========
A-SOUL
==========MARK_UP_OUTPUT==========
A-SOUL
1-A-SOUL
```

```
==========MARK_UP_OUTPUT==========
A-SOUL
1A-SOUL
==========MARK_UP_OUTPUT==========
A-SOUL

1A-SOUL
```