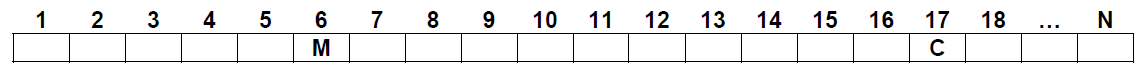
**Проектирование и реализация простых классов.**

**Объектно-ориентированные кошки-мышки**

Реализуйте объектно-ориентированную версию игры “кошки-мышки”. Кот и мышь находятся в ограниченном линейном пространство, которое состоит из **N** клеток; Например:



Каждая клетка имеет целочисленный номер. На рисунке выше мышь **М** находится в клетке **6**. А кот **C** находится в клетке **17**. Кот и мышь играют в игру. Каждый может двигаться, но не может покинуть последовательность клеток. Если кот и мышь оказываются в одной клетке после совершения хода, то мышь поймана и игра окончена.

Напишите программу для отслеживания перемещений кота и мыши, имея на входе информацию об их перемещениях. Каждый игрок (кот или мышь) перемещается между клетками. Ход задается указанием одного целого числа (например 4 или -7), указывающего, на какое количество клеток игрок должен перемещаться по горизонтали. Например, если кот сделал ход **-15** из показанного выше положения, то он окажется в клетке **2**. Обратите внимание, что в этом случае мышь не будет поймана.

Ни одному из игроков не разрешается перемещаться за пределы последовательности клеток. Если такой ход будет предпринят, игрок просто перемещается к противоположному концу последовательности. Представьте, что последовательность ячеек образует круг. Итак, если мышь сделала ход **-7** из своей позиции в клетке **6**, то она окажется в клетке **N-1**.

Учитывая начальные положения игроков, показанные выше, расстояние между котом и мышью равно **11**. Расстояния никогда не бывают отрицательными. Мы можем измерять расстояние, пройденное игроком с начала игры. Если кот сделал ходы 3, -5 и 10 от его начального положения, то он прошел бы общее расстояние **18**. Обратите внимание, что начальное положение не считается ходом.

**Формат входного файла и пример:**

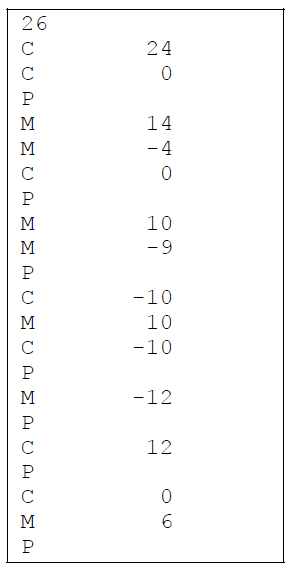
Ваша программа должна считывать входные данные из файла с именем ChaseData.txt. Первая строка входного файла указывает количество клеток, используемых для игры. Это значение является целым положительным числом и не превышает 10000. Клетки нумеруются, начиная с 1. Каждая следующая строка входных данных является командой к действию, представленной в одном из следующих форматов:

**M <integer>** // начальная позиция мыши или ход мыши

**C <integer>** // начальная позиция кота или ход кота

**P** // печать - команда вывода текущего состояния игры

Значения в каждой строке разделены пробелом. Все входные значения логически правильные. Пример входного файла:



**Что необходимо сделать:**

Написать программу, которая на основе данных входного файла:

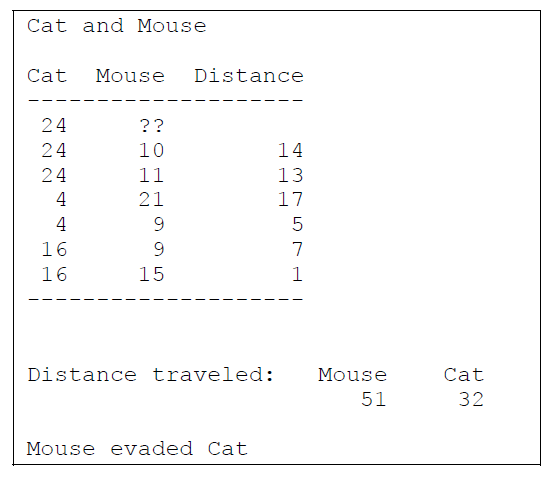
* определяет текущее положение кота и мыши
* определяет расстояние, которое разделяет кота и мышь (Distance)
* определяет когда (если это случится) кот ловит мышь
* вычисляет общее расстояние, пройденное каждым за время игры (Distance traveled).

Если кот ловит мышь, то обработка входного файла прекращается.

**Формат выходного файла и пример:**

* первая строка выходного файла должна содержать только заголовок «Cat and Mouse»;
* вторая строка должна быть пустой;
* третья строка содержит заголовки столбцов таблицы, которая будет выводиться далее (Cat, Mouse, Distance)
* четвертая строка - строка символов разделителей, например —-----------------;
* следующие строки файла - это результат выполнения команды **P** (печать), которая выводит местоположение кота и мыши и расстояние между ними (при условии, что оба находятся “в игре”);
* если кот или мышь еще “не в игре”, то вместо числового значения выводится “??”;
* после таблицы следует строка символов-разделителей;
* две пустые строки;
* в следующих двух строках выводится расстояние, пройденное игроками (показано ниже);
* пустая строка;
* последняя строка файла, содержит сообщение “Мышь ускользнула от кота - Mouse evaded Cat” или “Мышь поймана в клетке: <номер клетки> - Mouse caught at: <cell number>”

Программа должна записать выходные данные в файл PursuitLog.txt.



Протестируйте свое решение на предложенных файлах входных и выходных данных: 1.ChaseData.txt, 1.PursuitLog.txt, 2.ChaseData.txt, 2.PursuitLog.txt, 3.ChaseData.txt, 3.PursuitLog.txt.

**Требования к реализации:**

В **игру** играют два **игрока** (Кот и Мышь), находясь на поле, которое состоит из N клеток. **N** - размер игрового поля. У игрока есть имя, текущая позиция (номер клетки) и состояние. Состояние игрока можно описать следующими возможными значениями:

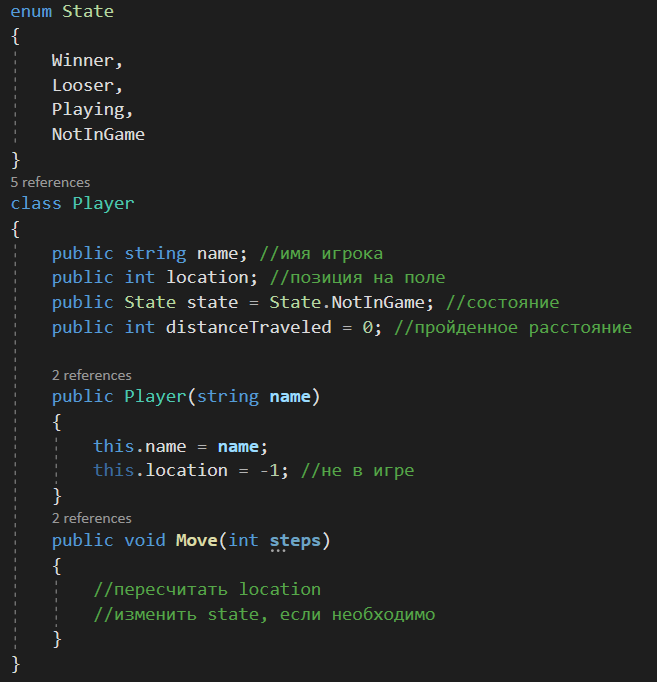
* Играет (Playing)
* Победитель (Winner)
* Проигравший (Loser)
* Не в игре (NotInGame) - позиция на поле еще не задана

Игрок может перемещаться (Move) на определенное количество клеток вперед или назад. Игра заключается в последовательном выполнении команд, записанных во входном файле ChaseData.txt. Возможны три типа команд: M, C и P (см. описание выше). Кот проиграл, если не сумел поймать мышь. Мышь проиграла, если была поймана котом. Результаты игры необходимо вывести на экран в соответствии с форматом, представленным в файлах PursuitLog.txt

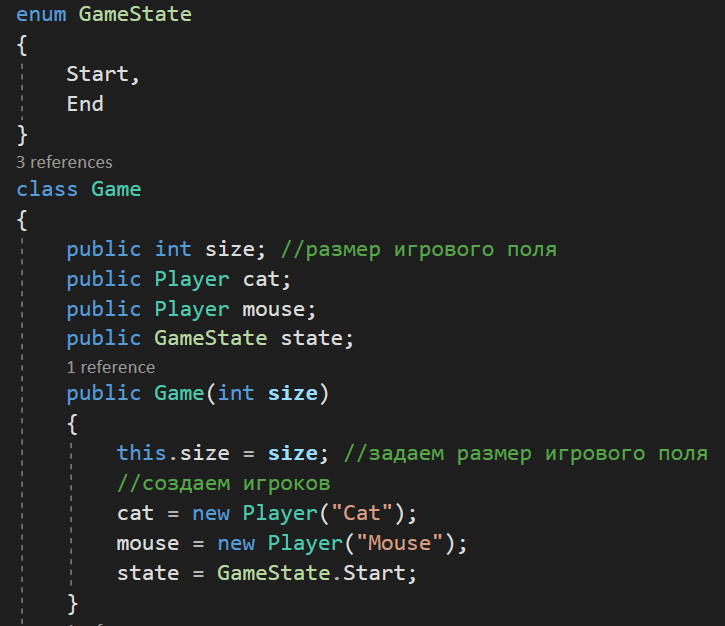
Условие задачи предполагает реализацию как минимум двух классов (Player и Game). Примерная модель реализации этих классов представлена ниже. Классы реализованы лишь частично. Вам предстоит завершить предложенную реализацию или предложить свою объектно-ориентированную модель игры. Классы должны содержать, как минимум:

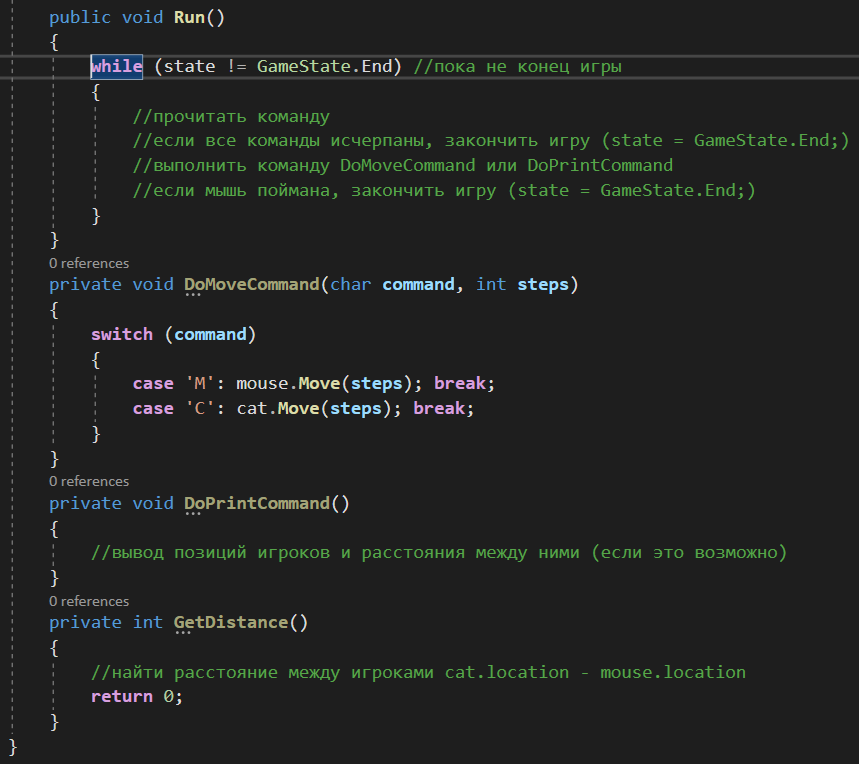
* поля, описывающие объект;
* конструктор, инициализирующий поля объекта начальными значениями;
* методы, определяющие поведение объекта.

**Класс Player**

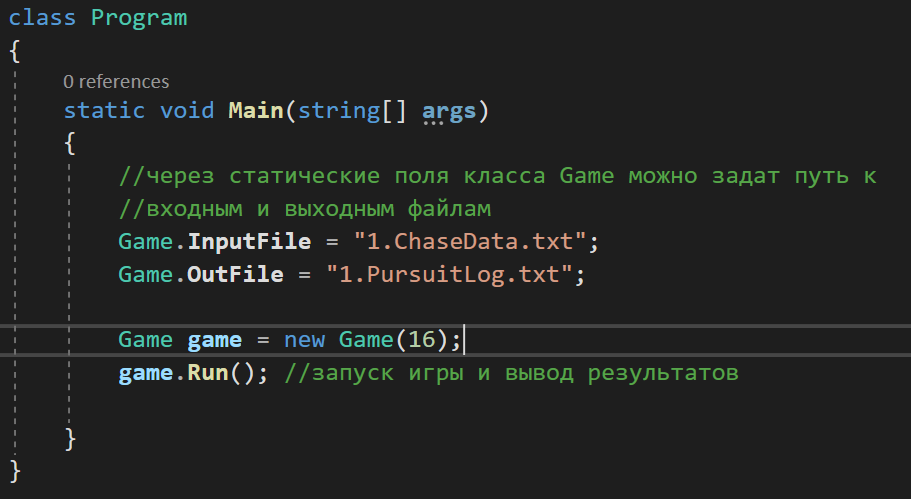
****

**Класс Game**





**Клиентский класс**

****