ТИП ШАШКА

Шашку характеризуют координаты xy (или номер клетки доски, что эквивалентно) – 6 бит, тип (шашка или дамка) – 1 бит, цвет – 1 бит, номер следующей и предыдущей шашки (каждая шашка эквивалентна номеру 1-12=> 4\*2 бит). Можно организовать ее структурой или запихнуть всю информацию в 1 инт.

Список шашек можно организовать 2 способами:

1) Шашки – обычный список из элементов (элемент = структура или инт).

2) Шашки – массив из 12 элементов. Тогда шашке определенного цвета ставим в соответствие ее номер. По номеру легко найти шашку, но при условии, что цвет известен, поэтому доску все равно заполняем указателями (на инт или структуру, лежащую в массиве). На основе массива легко организовать список. Черные и белые шашки по факту – двумерный массив, причем первый индекс – цвет, но этот двумерный массив можно организовать по-разному (например, одномерный массив из 2 элементов – готовых списков шашек).

ТИП ХОД (MOVE)

2 способа:

1) Ход состоит из номера шашки (или указателя на шашку), клетки, на которую шашка придет после выполнения хода, и съеденных шашек (вариант проще, но памяти займет немного больше, в теории, на 1 инт + если заводить статически массив ходов в рекурсии, то не понятно, какого размера).

2) Ход состоит из номера шашки (или указателя на шашку), клеток, на которые шашка может сходить, и съеденных шашек к каждой клетке хода (как реализовывали ранее).

Съеденные шашки можно организовать по-разному:

1) Выделить динамически массив интов в куче, в котором будут храниться номера шашек, и хранить указатель;

2) Поскольку номер шашки занимает всего 4 бита (в случае, если мы шашки храним в массиве), то можно запихнуть съеденные шашки максимум в 12\*4 бит, но это больше инта. Съеденные шашки могут храниться либо в long long, но размер этого типа не стандартизирован, либо в 2 интах, но тогда скорее всего придется писать класс, который будет обрабатывать их;

3) Хранить 12 бит (1 инт) с нулями и единицами, где каждый бит с номером n будет указывать, съедена ли шашка противоположного цвета с номером n.

С использованием варианта 3 хранения съеденных шашек и 1 способа организовать ход памяти будет затрачено 4+6+12=24 бита (1 инт) (может быть + 1 бит для уточнения цвета). Но представление будет нечитаемое, поэтому скорее всего есть смысл организовать класс, ну или макросами.

ТИП СПИСОК

Если создавать и тип ход, и тип шашка интом, то тогда список можно реализовать общий для обоих типов и агрегировать (тогда лучше писать один список шаблоном) или унаследовать (но я что-то не представляю как это сделать). Если оба структурой, то класс список будет выглядеть другим образом. Если по-разному реализовывать шашки и ход, то придется создавать два класса список, что не имеет смысла, лучше написать тогда отдельно список шашек и отдельно список ходов. Но это все при условии, что список ходов создается статически.

Но стоит ли ходы реализовывать списком? Можно ходы реализовать массивом, ведь мы ничего не удаляем. Но в таком случае либо мы фиксируем максимальный размер массива возможных ходов и заводим статически в рекурсии, либо заводим его в куче, либо глобальной колбасой. С другой стороны, если мы реализуем статический список ходов, а не массив ходов, то нам все равно так или иначе придется фиксировать размер списка, так что в списке ходов мало смысла + дополнительная память на связку.

Так что может быть есть смысл реализовать список шашек и все, а ходы массивом записывать?