ПОНГ

Были использованы библиотеки STL:

1. для ввода/вывода текстовой информации в консоли
2. в качестве опорного контейнера использовался двухмерный символьный вектор
3. для интерактивности использовался ввод с клавиатуры
4. перерисовка игрового поля происходит с задержкой времени

В классе MyGame все методы и часть членов-данных являются открытыми, к которым непосредственно происходит доступ через объект класса во время игры.

Член tempo задает изначальный темп игры. Используется в качестве стартового параметра в миллисекундах для перерисовки кадра (задержка).

Член autoBot позволяет при запуске игры выбрать режим игры с ботом (если true), иначе по умолчанию выбирается режим игры на 2 персоны.

Члены scoreLeft и scoreRight являются очками игроков (максимум 3, после чего текущая игра оканчивается).

Член desk является вектором векторов, состоящих из символьных констант. В качестве рамки игрового поля, шарика и ракеток использовались символы ASCII. Во время игры происходит переотрисовка экрана с новыми положениями символов в массиве (т.е. шарика и ракеток).

Члены leftPos и rightPos представляют верхние положения ракеток, используются для корректировки их движений.

Булевы переменные LTR, DIAG, UP: LTR определяет, что шарик может двигаться слева направо, пока не достигнет противоположной ракетки или границы, в этом случае меняется на противоположное значение. Аналогично DIAG означает возможность отбиваться мяча по диагонали, UP обозначает движение по диагонали именно вверх. Таким образом, переключаясь, мы получаем 6 возможных комбинаций перемещения по полотну шарика.

Челочисельные члены X и Y – координаты текущие шарика, необходимо их отслеживать для проверки условий на не/попадание по ракеткам.

Перечисление posBot является вспомогательным элементом для обозначение, в какую из 3 частей своего поля необходимо боту перепрыгнуть, чтоб отбить мяч. Ориентир идет на положение шарика за 5 клеток от ракетки. Ракетка имеет длину в 5 единиц. Всего вертикальных единиц (символов) 15. Таким образом, каждый из параметров перечислений отвечает за свою часть поля.

Метод fillDesk() забивает изначально вектор символами рамки, ракеток, шарика и пробелов.

Метод showDesk() показывает на экран отформатированный массив с дополненными пояснительными строками.

В методе padMove() при помощи функции \_kbhit() и \_getch() происходит считка символа, нажатого с клавиатуры для дальнейшего анализа и выбора реакции в цикле.

Управление в игре происходит следующим образом:

ЛЕВАЯ РАКЕТКА ВВЕРХ –> ‘A’

ЛЕВАЯ РАКЕТКА ВНИЗ–> ‘Z’

ЛЕВАЯ РАКЕТКА ВВЕРХ –> ‘K’

ЛЕВАЯ РАКЕТКА ВВЕРХ –> ‘M’

Для режима игры с ботом активна только левая ракетка.

Функция resetDesk() сбрасывает игру к начальным положениям ракеток и мяча после очередного пропуска шарика одним из игроков. При помощи функций rand() и srand() происходит рандомное новое положение шарика возле первой ракетки после каждого пропуска.

Функции геттеры собирают информацию про текущие положения ракеток и мяча.

Метод moveBot() принимает в качестве параметра один из вышеописанных элементов перечисления непосредственно для отрисовки ракетки бота, чтоб он имел возможность отбить шарик.

Метод botPlay() непосредственно реализует логику отбивания ботом шарика. Алгоритм имеет пропуски (прописаны не идеально все условия) для создания эффекта ошибки бота. Можно заменить этот алгоритм комбинацией полной версии и рандомной выборки.

Метод ballMove() непосредственно касается всех действий мяча, его перемещения (отрисовки) по игровому полотну.

Главная функция оформлена в виде цикла, который реагирует на нажатие клавиш:

‘b’ – ИГРА С БОТОМ

‘t’ – ИГРА НА 2 ИГРОКА

После завершения игры у игрока есть возможность продолжить игру нажатием ‘r’ или выйти из игры нажатием ‘q’. Это реализовано во внешнем цикле.