

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ НАУК**

**МИКРОПРОЕКТ НА ЯЗЫКЕ АССЕМБЛЕРА
Пояснительная записка**

**Выполнил студент группы БПИ 197
Глущенко Захар Сергеевич**

1. Текст задания

Задание: «Разработать программу, которая решает вопрос о нахождении пар перпендикулярных отрезков из $N=5$ отрезков, заданных координатами концевых точек».

2. Применяемые расчётные методы

В программе использовался следующий алгоритм поиска пар перпендикулярно пересекающихся отрезков:

1. Вложенным циклом ищем не повторяющуюся пару отрезков.
2. Через скалярное произведение их векторов проверяем перпендикулярность пары отрезков.
3. Проверяем пересечение: через псевдоскалярное произведение проверяем положение точек второго отрезка относительно прямой первого отрезка и положение точек первого отрезка относительно прямой второго отрезка.
4. Если на третьем пункте подтверждается пересечение отрезков (Псевдоскалярные произведения прямой на точки должны дать разные знаки, а значит и численное произведение двух скалярных произведений между собой должно быть меньше 0), то выводим точки пары отрезков в консоль.
5. Если ни одной подходящей пары не было найдено, выводится сообщение об отсутствии пар перпендикулярных пересекающихся отрезков.

Использованные переменные:

- Массивы символов для ввода вывода текста
 - inputText1 - Подсказка для пользователя 1
 - inputText2 - Подсказка для пользователя 2
 - inputText3 - Подсказка для пользователя 3
 - input - Содержит формат ввода чисел
 - outputres - Содержит формат вывода результат
 - notResult - Содержит информацию об отсутствие результатов
 - endprog - Сообщает о завершении программы
- Массивы для хранения введенных координат точек
 - rx - Содержит 10 точек пяти отрезков. Точки в массиве расположены парой из двух координат по оси x ($rx[i]$, $rx[i + 1]$ - координаты x двух точек отрезка).
 - ry - Аналогично rx содержит 10 точек координат y пяти отрезков.
- Переменные для промежуточного хранения результата вычисления координатов векторов
 - x1
 - x2
 - y1
 - y2
- Переменные для хранения промежуточных вычислений
 - tmp1
 - tmp2
- Переменные для хранения промежуточных значений знака поворота вектора при псевдоскалярном умножении
 - sign1
 - sign2
- Переменные для индексирования по массивам и работе с циклами
 - index
 - jindex
- Переменные для хранения количества найденных пар
 - count

Макросы, использованные в программе:

- `macro make_vector op1, op2, op3`
Создает координату вектора по двум точкам. Результат сохраняется в `op3` (`op3 = op2 - op1`)
- `macro make_vector2 vec, ind, res`
Создает координату вектора по двум точкам используя массив `vec` и индекс текущего отрезка. Результат сохраняется в `res` (`res = vec[ind] - vec[ind + 1]`)
- `macro multiply_nums a, b, res`
Совершает знаковое умножение двух чисел и возвращает результат в `res` (`res = a * b`)
- `macro GetTurnSign ax1, ax2, ay1, ay2, bx1, by1`
Берет значение поворота вектора с его знаком (получаем вектор (`x1 = ax2 - ax1`, `y1 = ay2 - ay1`) и (`x2 = bx1 - ax1`, `y2 = by1 - ay1`), а затем при помощи псевдоскалярного произведения (`x1 * y2 - x2 * y1`) получаем знак поворота). Результат вычислений записывается в регистр `eax`
- `macro IsIntersect ax1, ax2, ay1, ay2, bx1, bx2, by1, by2`
Выводит переданную пару отрезков, если они пересекаются.
 - При помощи двух макросов `GetTurnSign`, получаем знак поворота для двух точек второго отрезка относительно первого отрезка.
 - Повторяем предыдущий пункт для двух точек первого отрезка относительно второго отрезка
 - Если на первом или втором шаге точки находятся на одной стороне относительно проверяемого отрезка, выходим из макроса
 - Иначе выводим переданную пару отрезков

Также в программе использовались функции из библиотек `kernel32.dll` и `msvcrt.dll`:

- `ExitProcess`
- `printf`
- `scanf`
- `_getch`

3. Список используемых источников

- 1) [Базовые соотношения и алгоритмы геометрии](#)
- 2) [Псевдоскалярное произведение](#)
- 3) Аблязов Р. 3. Программирование на ассемблере на платформе x86-64. – М.: ДМК Пресс, 2011. – 304 с.: ил

4. Текст программы

Текст программы можно найти в файле **Глущенко_ТП.pdf**, находящемся в одной директории с данным документом.

5. Описание области допустимых значений входных параметров

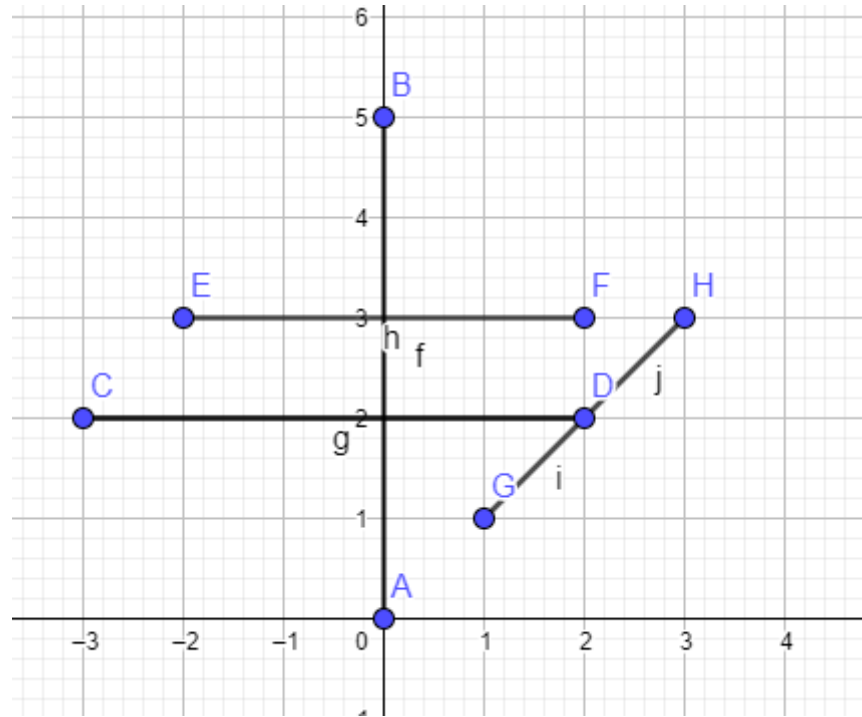
В качестве входных параметров пользователем вводится 5 строк, по 4 числа на каждой, обозначающие координаты в формате `x1 y1 x2 y2`.

6. Тестовые примеры

- 1) Пользователем вводится числа, находящийся в рамках допустимых значений и в заданном формате.

Пример1:

0 0 0 5
-3 2 2 2
1 1 2 2
2 2 3 3
-2 3 2 3



Отрезки визуализированные на [сайте](#) для демонстрации их расположения

```
C:\Users\mizar\Desktop\Assembler_hw\miniProject1\m1.EXE
Input five lines in format: x1 y1 x2 y2
Example: We have line at point (1, 0) and point (0, 1)
As a result, we will input: 1 0 0 1
0 0 0 5
-3 2 2 2
1 1 2 2
2 2 3 3
-2 3 2 3
Line1: ({0, 0}, {0, 5}); Line2: ({-3, 2}, {2, 2})
Line1: ({0, 0}, {0, 5}); Line2: ({-2, 3}, {2, 3})
Program is end
```

Результат будет выведен в формате “Line1 ({x1,y1},{x2,y2}); Line2({x1,y1},{x2,y2})”,
где x1 y1 x2 y2 – координаты для соответствующего отрезка

Пример2:

1 1 1 1

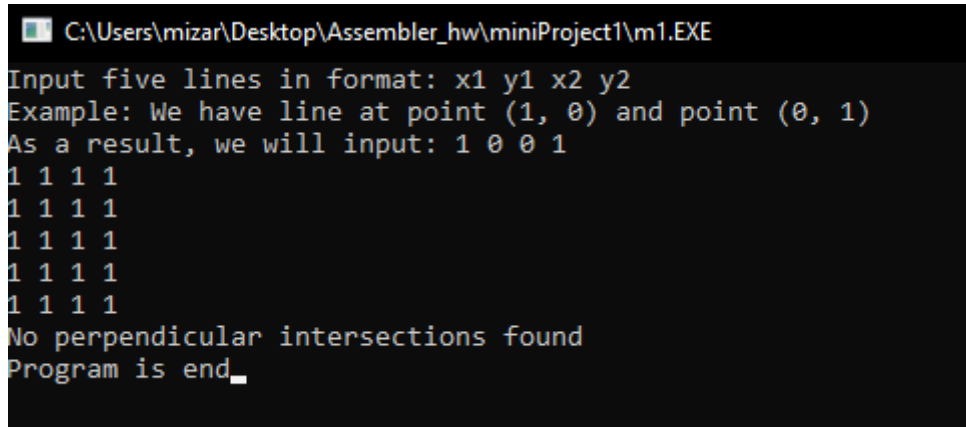
1 1 1 1

1 1 1 1

1 1 1 1

1 1 1 1

Если перпендикулярных отрезков нету, то результат работы программы – сообщение об отсутствии таковых.



```
C:\Users\mizar\Desktop\Assembler_hw\miniProject1\m1.EXE
Input five lines in format: x1 y1 x2 y2
Example: We have line at point (1, 0) and point (0, 1)
As a result, we will input: 1 0 0 1
1 1 1 1
1 1 1 1
1 1 1 1
1 1 1 1
1 1 1 1
No perpendicular intersections found
Program is end.
```