

ЗАНЯТИЕ 22. ТОЧКА И МНОГОУГОЛЬНИК

1 Перечисляемый тип данных

Перечисляемый тип данных задаётся перечислением тех значений, которые он может принимать. Никакие другие значения переменной перечисляемого типа присвоить нельзя. Каждое значение задаётся некоторым идентификатором и располагается в списке, разделённом запятыми и заключённом в круглые скобки. Например

```
Type TColor=(red,green,blue);
```

Переменная типа TColor, определённого выше, может принимать только значения red, green и blue. Перечисляемые типы данных используют для удобочитаемости программы. Для внутреннего хранения компилятор преобразует каждый из заданных идентификаторов в целое число. Первое значение соответствует значению 0, второе 1, третье 2 и т.д. Однако, это не значит, что с переменными такого типа можно осуществлять математические операции. Какое именно число поставлено в соответствие какому идентификатору можно узнать при помощи функции Ord. Например, в примере выше Ord(blue) даст значение 2. Если есть желание изменить внутреннее представление перечисляемого типа, можно написать

```
Type TColor2=(red,green=10,blue);
```

Тогда Ord(red) будет равно 0, Ord(green) — 10, Ord(blue) — 11. То есть представление некоторых идентификаторов можно указать явно. Тогда представление следующего идентификатора будет на 1 больше, если оно не задано явно.

Разрешенными для перечисляемых типов являются операции Pred и Succ. Функцию Succ применяют для получения следующего значения переменной, а функцию Pred — предыдущего. Например, если переменная x имеет значение green, то Succ(x) равно blue, а Pred(x) даёт значение red. Нельзя вычислить Pred для наименьшего (первого) значения перечисляемого типа. Не определена также операция Succ для наибольшего (последнего) значения.

2 Попадание точки в многоугольник

Как и в занятии 21, говоря далее о многоугольниках, всегда будем иметь ввиду простой плоский многоугольник.

Задача 1. Заданы координаты вершин $A_1 A_2 \dots A_n$ многоугольника в порядке их обхода по или против часовой стрелки и координаты некоторой точки B . Определить, где находится точка B относительно этого многоугольника: внутри, снаружи или на его границе.

Нам уже встречалась похожая задача¹. Но тогда многоугольник был выпуклым. Рассмотрим теперь случай, включающий также и невыпуклые многоугольники.

Алгоритм решения этой задачи можно найти в работах² Кормена [2, стр. 1055] и Андреевой [1].

¹См. занятие 20 задачу 6 из заданий для самостоятельного решения

²См. список литературы

Из точки B проводится произвольный луч и определяется, сколько раз он пересекает границу многоугольника. Если количество пересечений нечётно и сама точка B не лежит на границе многоугольника, то она лежит во внутренней его части.

Без ограничения общности можно считать, что луч из точки B направлен параллельно оси x в сторону её положительного направления. При традиционном выборе направления осей это значит, что луч направлен направо. Обозначим этот луч BK .

На рисунке 1 точка B лежит вне многоугольника, и луч BK пересекает границу 4 раза. Он мог делать это любое чётное число раз, в том числе и ни разу. На рисунке 2 точка B расположена внутри многоугольника, и луч BK пересекает его границу 3 раза.

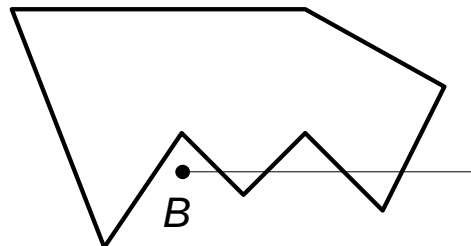


Рис. 1: Точка вне многоугольника.

2.1 Подсчёт количества пересечений границы лучом

Однако не всегда легко подсчитать количество пересечений границы многоугольника лучом BK . Если стороны многоугольника могут пересекаться лучом каждая не более чем в одной точке, отличной от вершин, то проблем нет. Рисунки 1 и 2 иллюстрируют именно эти случаи. Трудности возникают, когда

1. луч BK пересекает вершину многоугольника (см. рис. 3)

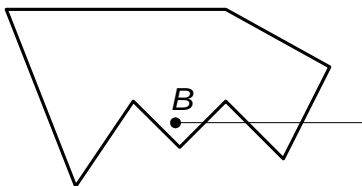


Рис. 2: Точка внутри многоугольника

2. сторона многоугольника полностью принадлежит лучу BK (см. рис. 4)

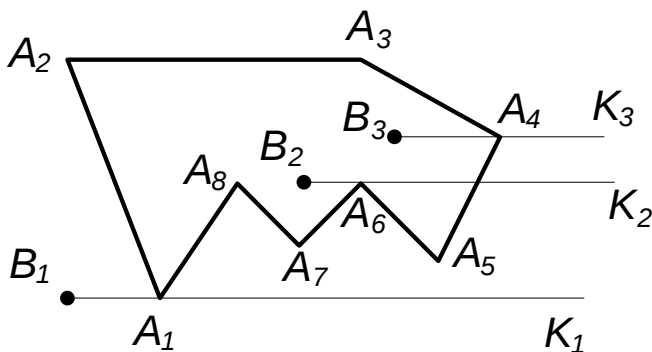
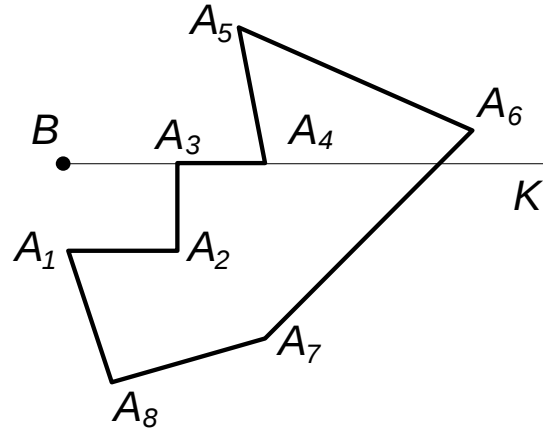


Рис. 3: Луч пересекает вершину

В этих случаях пересечение луча имеет место сразу с несколькими смежными сторонами многоугольника. Будем подсчитывать пересечения луча BK именно со сторонами многоугольника.

2.1.1 Луч пересекает вершину многоугольника

Рассмотрим случай 1. Если луч просто касается границы многоугольника в его вершине, как для луча B_1K_1 и вершины A_1 на рис. 3, то такое пересечение с *границей* учитывать не надо. Этого можно достичь или учитывая пересечения луча с обеими сторонами A_1A_2 и A_1A_8 , или оба пересечения со сторонами не учитывая. Это не изменит *чётности* количества пересечений границы, а только она имеет значение для ответа на вопрос задачи. Такой же подход применим и для луча B_2K_2 с вершиной A_6 . Здесь луч B_2K_2 пересекается со сторонами A_5A_6 и A_6A_7 .



Пусть теперь BK действительно пересекает границу многоугольника в его вершине, в том смысле, что часть луча принадлежит многоугольнику, а другая часть нет (рис. 3, вершина A_4 и луч B_3K_3). В этом случае чётность пересечений границы многоугольника надо поменять. Этого можно достичь, *учитывая пересечения луча с верхними концами сторон многоугольника и не учитывая с нижними*. То есть пересечение B_3K_3 с A_4A_5 учитывать будем, а с A_3A_4 — нет. Заметим, что если придерживаться этого соглашения, то будет корректно обрабатываться и рассмотренный выше случай касания. В вершине A_1 оба пересечения луча со сторонами будут проигнорированы, а в вершине A_6 — оба учтены.

Рис. 4: Сторона многоугольника полностью на луче

2.1.2 Сторона многоугольника принадлежит лучу

В случае 2 пересечение луча с полностью принадлежащей лучу стороной (но не с другими!) можно просто игнорировать. На рис. 4 можно игнорировать пересечение луча BK со стороной A_3A_4 . Пересечения со сторонами A_2A_3 и A_4A_5 будем рассматривать, учитывая принятую договорённость. Итого будут учтены 2 пересечения BK с границей многоугольника: в точке A_3 и на отрезке A_6A_7 . Чётное число пересечений соответствует положению точки B за пределами многоугольника.

2.2 Определение принадлежности точки отрезку

Пересечения луча с границей многоугольника надо считать, если удалось выяснить, что точка B не лежит на этой границе. А чтобы это проверить, надо решить задачу принадлежности точки отрезку.

Задача 2. Определить, принадлежит ли точка C отрезку AB .

Проведём из точки C векторы $\overrightarrow{CA}$ и $\overrightarrow{CB}$ в концы отрезка AB . Если точка C принадлежит отрезку AB , то либо один из этих векторов равен нулевому вектору, либо угол между $\overrightarrow{CA}$ и $\overrightarrow{CB}$ равен π , т.е. 180 градусов. Дальше можно идти несколькими путями.

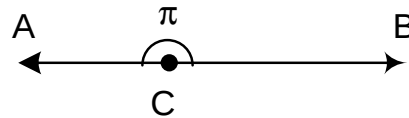


Рис. 5: Принадлежность точки отрезку

2.2.1 Прямое использование скалярного произведения векторов

Допущение 1. Пусть точка C не совпадает ни с одним из концов отрезка AB .

Вычислим скалярное произведение

$$\overrightarrow{CA} \cdot \overrightarrow{CB} = CA \cdot CB \cdot \cos \alpha \quad (1)$$

где α — угол между $\overrightarrow{CA}$ и $\overrightarrow{CB}$. Если точка C является внутренней точкой отрезка AB , то $\alpha = \pi$, $\cos \pi = -1$ и

$$\overrightarrow{CA} \cdot \overrightarrow{CB} = -CA \cdot CB \quad (2)$$

Более того, как следует из формулы (1), ни для какого другого значения угла $0 \leq \alpha \leq \pi$ равенство (2) выполняться не может, если только выполняется допущение 1.

Таким образом, проверяя выполнение допущения 1 и равенства (2), можно определить принадлежность точки отрезку.

Но тут может возникнуть одно неудобство. Даже если координаты всех точек — целые числа, при использовании такого подхода нам не удастся избежать вычислений с вещественными числами. Ведь длины отрезков вычисляются через проекции вектора на оси координат с использованием теоремы Пифагора:

$$CA = \sqrt{CA_x^2 + CA_y^2}$$

2.2.2 Использование скалярного и векторного произведений

Вычислим векторное произведение $\overrightarrow{CA} \times \overrightarrow{CB}$. Его равенство $\vec{0}$ означает, что либо не выполняется допущение 1, либо векторы $\overrightarrow{CA}$ и $\overrightarrow{CB}$ параллельны. В обоих случаях получаем, что точки A , B и C лежат на одной прямой. Вопрос только в том, не лежит ли весь отрезок AB по одну сторону от точки C .

На рис. 6 показан один из таких случаев.

Возможно также, что точка C окажется правее отрезка AB . Чтобы распознать эти случаи, используем скалярное произведение $\overrightarrow{CA} \cdot \overrightarrow{CB}$.

Как следует из формулы (1), оно положительно тогда и только тогда, когда точки A и B лежат по одну сторону от C и не совпадают с ней.

Значит, точка C принадлежит отрезку AB тогда и только тогда, когда

$$\overrightarrow{CA} \times \overrightarrow{CB} = \vec{0}$$

и

$$\overrightarrow{CA} \cdot \overrightarrow{CB} \leq 0$$

Заметим также, что если координаты точек — целые числа, то скалярное произведение

$$\overrightarrow{CA} \cdot \overrightarrow{CB} = CA_x \cdot CB_x + CA_y \cdot CB_y \quad (3)$$

тоже целочисленно.

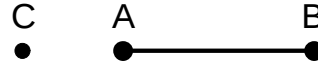


Рис. 6: Отрезок по одну сторону от точки

2.3 Определение пересечения отрезка лучом

Подзадачу определения пересечения отрезка лучом BK можно свести к задаче пересечения двух отрезков. Для этого достаточно выбрать точку K так далеко, чтобы её абсцисса была больше абсциссы любой из вершин многоугольника и больше абсциссы точки B .

Задача 3. Определить, имеет ли место пересечение отрезка BK , параллельного оси абсцисс, с отрезком P_1P_2 , если абсцисса точки K больше абсцисс всех остальных точек. Если да, то какой из случаев реализуется:

1. отрезок P_1P_2 пересечён в точке P_1 ;
2. отрезок P_1P_2 пересечён в точке P_2 ;
3. отрезок P_1P_2 пересечён во внутренней точке;
4. отрезок P_1P_2 полностью содержится в отрезке BK .

Учитывая, что отрезок BK ориентирован строго вдоль оси x и, кроме того, абсцисса точки K строго больше абсцисс всех других точек, эта задача проще, чем задача 3 занятия 20. Проблема, обозначенная на рис. 10 указанного занятия, решается теперь намного проще. Пересечения отрезков, лежащих на одной прямой, не будет, если точка B лежит правее отрезка P_1P_2 , т.е. $x_B > x_{P_1}$ и $x_B > x_{P_2}$.

В остальном подход остаётся тем же.

Если прямая BK пересекает отрезок P_1P_2 и прямая P_1P_2 пересекает отрезок BK , то отрезки пересекаются, иначе нет. При этом понадобится ещё отслеживать случай попадания концов отрезка P_1P_2 на отрезок BK .

Введём обозначения.

$$\begin{aligned}\vec{c} &= \overrightarrow{BP_1} \times \overrightarrow{BK} \\ \vec{d} &= \overrightarrow{BP_2} \times \overrightarrow{BK} \\ \vec{e} &= \overrightarrow{P_1B} \times \overrightarrow{P_1P_2} \\ \vec{f} &= \overrightarrow{P_1K} \times \overrightarrow{P_1P_2}\end{aligned}$$

Получается следующий алгоритм.

Алгоритм 1. *Определение пересечения стороны многоугольника лучом*

Если $\vec{c} \uparrow\uparrow \vec{d}$ или $\vec{e} \uparrow\uparrow \vec{f}$, то отрезки не пересекаются³.

Иначе если $c = 0$ и $d = 0$, то имеем случай 4 из условия задачи 3.

Иначе если $\vec{c} = \vec{0}$, то имеем случай 1.

Иначе если $\vec{d} = \vec{0}$, то имеем случай 2.

В противном случае имеет место случай 3.

2.4 Описание алгоритма

При решении задачи 1 удобно нумеровать вершины многоугольника не с единицы, а с нуля. Тогда легко будет учесть, что при обходе вершин после последней вершины должна идти первая. Для этого достаточно считать, что за вершиной с индексом i следует вершина с индексом $(i + 1) \bmod N$, где N — количество вершин многоугольника.

Пусть функция

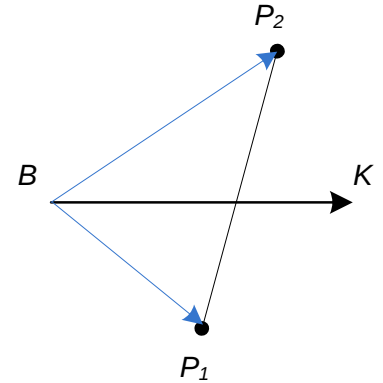


Рис. 7: Пересечение отрезков

³Знак $\uparrow\uparrow$ означает сонаправленность векторов.

```
Function PointInSegment(const A,B,C : TPoint) : Boolean;
```

определяет, принадлежит ли точка C отрезку AB , т.е. решает задачу 2. Для её реализации используем подход, изложенный в пункте 2.2.2.

Пусть функция

```
Function SegIntersect(const p1,p2,b,k : TPoint) : TIntersect;
```

есть реализация алгоритма 1.

В реализации последней функции использован перечисляемый тип данных

```
TIntersect = (S_NONE,S_P1,S_P2,S_INNER,S_BOTH);
```

Значение S_NONE соответствует отсутствию пересечения, S_P1 — пересечению только в точке P_1 , S_P2 — только в точке P_2 , S_INNER — пересечению во внутренней точке, S_BOTH — случаю, когда весь отрезок P_1P_2 принадлежит отрезку BK .

Тогда для решения задачи 1

1. Обходим все стороны многоугольника и проверяем, не лежит ли точка B на одной из сторон. Если лежит, решение найдено.
2. Находим точку K как точку, абсцисса которой на 1 больше абсциссы самой правой точки из всех точек, о которых идёт речь в задаче. $y_K = y_B$.
3. Счётчик $count := 0$
4. Обходим все стороны многоугольника и
 - (a) вычисляем $r := \text{SegIntersect}$ для каждой из них.
 - (b) Если $r = S_INNER$, или пересечение в первой вершине и первая вершина выше второй, или пересечение во второй вершине и вторая вершина выше первой, то $count := count + 1$
5. Если $count$ чётно, то точка B лежит за пределами многоугольника, иначе — внутри многоугольника.

2.5 Программа

Программу решения задачи 1 пишем в предположении, что координаты всех точек — целые числа, для хранения которых достаточно 32-битового целого числа. Входные данные читаются из файла `input.txt` в следующем порядке: количество вершин многоугольника N , в следующих N строках координаты x и y каждой из вершин, далее координаты точки B .

Текст программы на FreePascal смотрите в файле dolschool5.narod.ru/dist/pt_inreg.pas

Задания для самостоятельного выполнения

1. Можно ли в задаче 1 изменить соглашение об учёте пересечения луча и сторон многоугольника на следующее: учитывать пересечения луча с нижними концами сторон многоугольника и не учитывать с верхними? Потребуется ли после этого существенная переработка алгоритма?
2. Попробуйте несколько видоизменить способ определения принадлежности точки отрезку, описанный в пункте 2.2.2, так, чтобы не использовать скалярное произведение. Ориентируйтесь на знаки проекций векторов. Не забудьте, что абсциссы некоторых точек могут оказаться одинаковыми.
3. Напишите программу решения задачи 1, проводя не горизонтальный, а вертикальный луч.

Список литературы

- [1] Андреева Е.В., Егоров Ю.Е. Вычислительная геометрия на плоскости. //Информатика №39–44/2002.
- [2] Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: Построение и анализ, 2-е издание. : Пер. с англ. — М.: Издательский дом «Вильямс», 2010.
- [3] Фаронов В.В. Турбо Паскаль 7.0. Начальный курс. Учебное пособие. — М.: «Нолидж», 1998.