|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Фундаментальные науки

КАФЕДРА Прикладная математика

**ОТЧЕТ ПО ПРАКТИКЕ**

Студент Климов Олег Дмитриевич

*фамилия, имя, отчество*

Группа ФН2-31Б

Тип практики: Ознакомительная практика

Название предприятия: Научно-учебный комплекс «Фундаментальные науки»   
 МГТУ им. Н.Э. Баумана

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Климов О.Д.

*подпись, дата фамилия и.о.*

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Марчевский И.К.

*подпись, дата фамилия и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2022 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

# (МГТУ им. Н.Э. Баумана)

Кафедра «Прикладная математика»

**З А Д А Н И Е**

**на прохождение ознакомительной практики**

на предприятии Научно-учебный комплекс «Фундаментальные науки»   
 МГТУ им. Н.Э. Баумана

Студент Климов Олег Дмитриевич

*фамилия, имя, отчество*

Во время прохождения ознакомительной практики студент должен

1. Изучить на практике основные возможности языка программирования С++ и систем компьютерной алгебры, закрепить знания и умения,

полученные в курсах «Введение в информационные технологии»,

«Информационные технологии профессиональной деятельности».

1. Изучить способы реализации методов определения локализации точки в многоугольнике.
2. Реализовать алгоритм определения положения точки относительно многоугольника на плоскости и в пространстве.

Дата выдачи задания «16» сентября 2022 г.

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Марчевский И.К.

*подпись, дата фамилия и.о.*

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Климов О.Д.

*подпись, дата фамилия и.о.*

**Содержание**

[Задание 4](#_Toc60234685)

[Введение 5](#_Toc60234686)

[1. Задача о принадлежности точки многоугольнику 6](#_Toc60234687)

[2. Трассировка луча 8](#_Toc60234688)

[2.1. Описание метода 8](#_Toc60234689)

[2.2. Реализация метода 9](#_Toc60234690)

[3. Метод суммирования углов 10](#_Toc60234691)

[3.1. Описание метода 10](#_Toc60234692)

[3.2. Реализация метода 11](#_Toc60234693)

4. Метод ближней точки и нормали…………………………………………….12

4.1 Описание метода…………………………………………………………..12

4.2. Реализация метода………………………………………………………..14

5. Сравнительная таблица методов …………………………………………….15

[6. Задача о принадлежности точки в произвольной плоскости 16](#_Toc60234694)

[Заключение 17](#_Toc60234695)

[Список литературы 18](#_Toc60234696)

# Задание

Плоский многоугольник задан координатами своих вершин, и заданы координаты некоторого количества точек.

Определить, находится ли данные точки внутри или снаружи многоугольника. Реализовать не менее трех различных алгоритмов. При необходимости можно считать, что известны координаты некоторой точки, гарантировано лежащей внутри многоугольника. Рассмотреть 2 случая:

а) многоугольник лежит в плоскости Oxy;

б) многоугольник лежит в произвольной плоскости

Структура исходного файла данных:

n << количество исследуемых точек

x1 y1 << координаты первой исследуемой точки

…

xn yn << координатыn-й исследуемой точки

p << количество углов многоугольника

x1 y1 << координаты первого угла многоугольника

…

xp yp << координаты p-го угла многоугольника

x0 y0 << координаты точки, лежащей внутри многоугольника

Структура файла результата:

m << количество точек, лежащих внутри многоугольника

x1 y1 << координаты первой точки

…

xm ym << координаты m-й точки

Для пункта б) в файлы исходных данных и результата очевидным образом добавляется z-я координата всех точек.

# Введение

Основной целью ознакомительной практики 3-го семестра, входящей в учебный план подготовки бакалавров по направлению 01.03.04 – Прикладная математика, является знакомство с особенностями осуществления деятельности в рамках выбранного направления подготовки и получение навыков применения теоретических знаний в практической деятельности. В ранее пройденном курсе «Введение в специальность» произошло общее знакомство с возможными направлениями деятельности специалистов в области прикладной математики и получен опыт оформления работ (реферата), который полезен при оформлении отчета по практике. В рамках освоенного курса «Введение в информационные технологии» изучены основные возможности языка программирования С++ и сформированы базовые умения в области программирования на С++. Задачей практики является закрепление соответствующих знаний и умений и овладение навыками разработки программ на языке С++, реализующих заданные алгоритмы. Кроме того, практика предполагает формирование умений работы с системами компьютерной алгебры и уяснение различий в принципах построения алгоритмов решения задач при их реализации на языках программирования высокого уровня (к которым относится язык С++) и на языках функционального программирования (реализуемых системами компьютерной алгебры).

# 1. Задача о принадлежности точки многоугольнику

В области вычислительной геометрии известна задача о взаимном расположении точки относительно многоугольника. Формулировка этой задачи следующая: «Пусть на плоскости или в пространстве задаётся многоугольник координатами своих N вершин в порядке обхода по контуру. Для заданной своими координатами точки необходимо определить принадлежит ли она многоугольнику». Многоугольник может быть, как выпуклым, так и невыпуклым. Обычно рассматриваются простые многоугольники, то есть без самопересечений.

Для выпуклого многоугольника задача упрощается, так как он обладает некоторыми особенностями. Например, одной из особенностей является то, что отрезок из двух любых точек, принадлежащих многоугольнику, не пересекает сторон многоугольника. Также, выпуклый многоугольник можно легко разбить на треугольники и задачу можно свести к рассмотрению принадлежности точки к одному из треугольников, что гораздо легче.

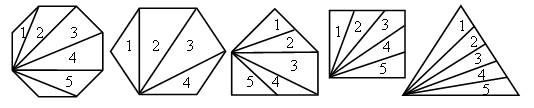


рис. 1. Выпуклый многоугольник и разбиение его на треугольники

Для невыпуклого многоугольника задача немного усложнится, но при применении определённых алгоритмов, эта задача может быть решена.

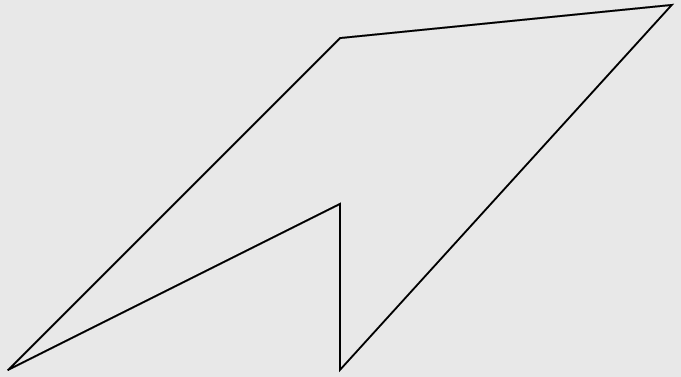


рис. 2. Невыпуклый многоугольник

Случай самопересекающегося многоугольника является самым сложным. В этом случае разные способы определения принадлежности точки многоугольнику могут привести к разным результатам.

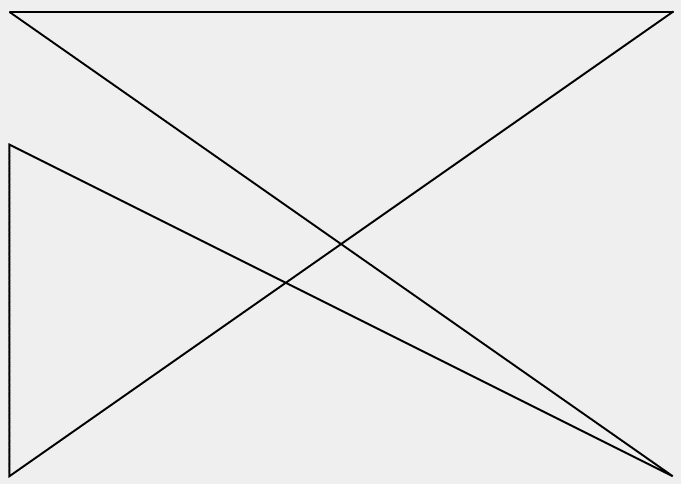


рис. 3. Самопересекающийся многоугольник

Далее все методы будут описаны для многоугольника, который задан набором точек и порядком в котором они соединяются, многоугольник замкнутый, и в общем случае ребра не пересекаются друг с другом.

# 2. Метод №1: Трассировка луча

## **2.1. Описание метода**

Этот метод является довольно популярным в мире графики и игр. Алгоритм этого метода можно описать следующим образом:

1. Из тестируемой точки выпускаем луч либо в заранее заданном (можно в произвольном) направлении.
2. Считаем количество пересечений с многоугольником.
3. Если количество пересечений четное, тогда точка находится вне многоугольника, а если нечетное, то внутри.

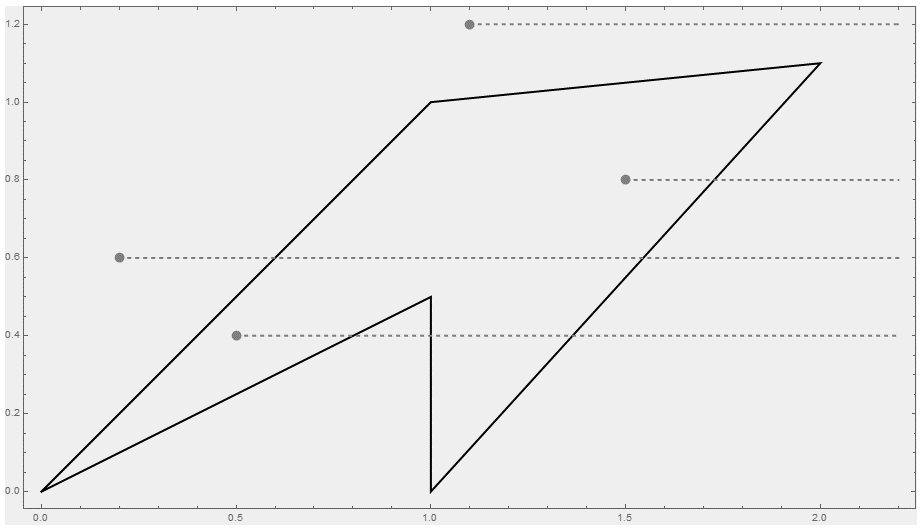
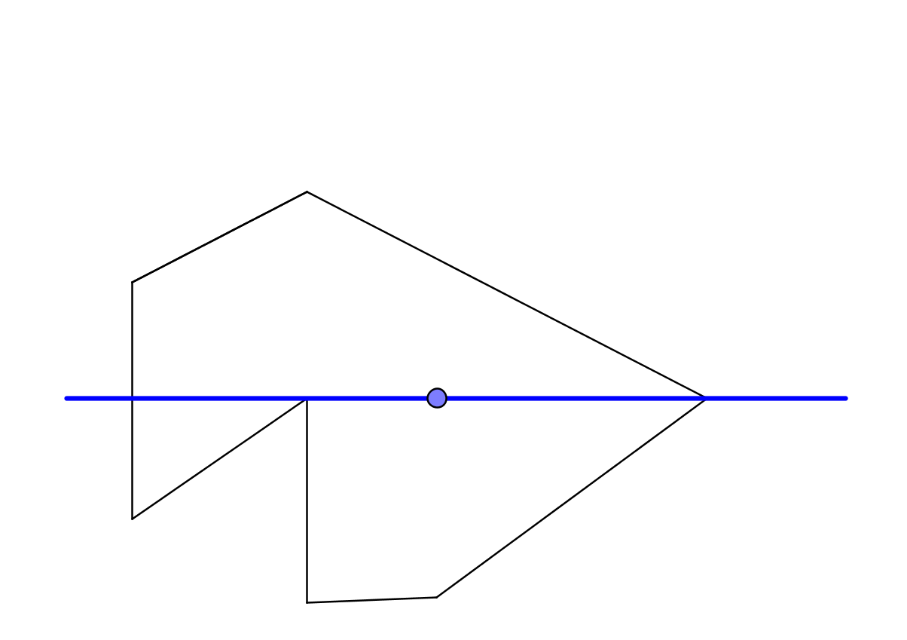


рис. 4. Трассировка луча

Главным достоинством этого метода является его простота. Недостаток метода заключается в том, что могут возникнуть исключительные ситуации, которые придется усмотреть. Например, когда луч пересекает вершину многоугольника или еще хуже - его сторону. Сложность алгоритма O(log N), где N ­­- кол-во вершин многоугольника.



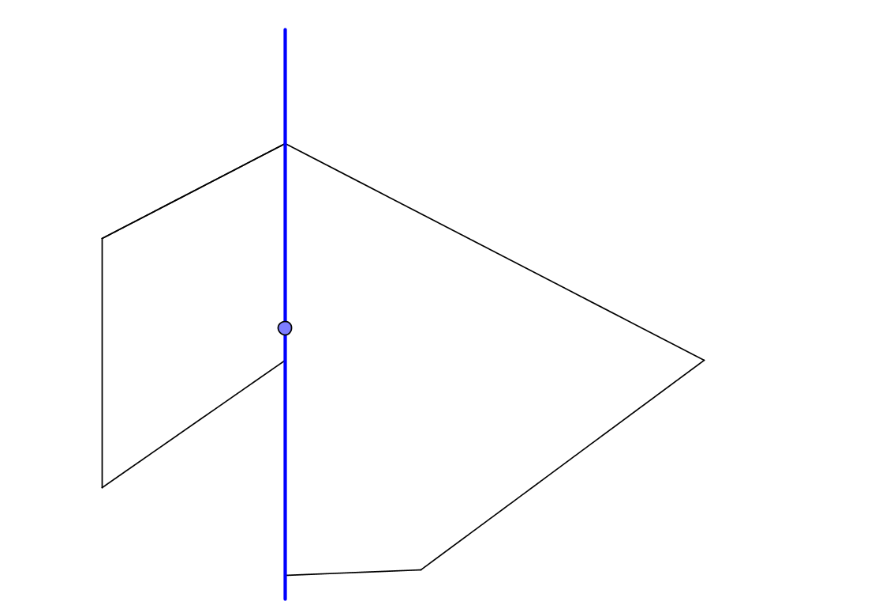


рис. 5. Возможные исключения

## **2.2. Реализация метода**

Необходимо организовать цикл для обхода всех вершин. Пусть S - целочисленная переменная принимает начальное значение, равное 1 . В теле цикла вычисляется произведение:

*S\*= met1Points (V[i], V[i+1], P)*,

Где V[i]­, V[i+1] – это i-ая и i+1-ая вершина многоугольника, а P ­– точка, которую мы исследуем на принадлежность многоугольнику, а луч, выпускаемы из точки, выпускается по координате Х вправо.

Функция *met1Points*, которая принимает в качестве аргумента сторону многоугольника (вершины i и i+1) и исследуемой точки.

Результатом этой функции будет 0, если исследуемая точка лежит на стороне или вершине. Если выпущенный луч пересекает сторону, результатом будет -1, если совпадает с ней, то функция возвратит 1. Если функция проходит через вершину i+1, и вершина i расположена ниже чем вершина i+1, то функция возвратит -1. Если функция проходит через вершину i, и вершина i+1 расположена ниже чем вершина i, то функция возвратит -1. В остальных случаях функция вернёт значение 1.

Таким образом, если значение s будет 1, то точка находится снаружи многоугольника, если с противоположным знаком, то точка внутри многоугольника.

Преимуществом реализации в системе компьютерной алгебры «Wolfram Mathematica 13.1» является более удобное использование списков, как векторов, нежели структур в с++, а также функции Intersection[n], которая позволяет найти общие элементы в двух списках, с помощью чего можно найти пересечение луча и стороны. Еще стоит отметить простую возможность визуализации.

# 3. Метод №2: Суммирование углов

## **3.1. Описание метода**

Ещё одним простым решением для задачи о локализации точки относительно многоугольника является метод суммирования углов. Суть метода в том, что необходимо обойти все вершины многоугольника и просуммировать все углы, которые образуются при переходе от одной вершины к другой, относительно исследуемой точке. Если поворот происходит против часовой стрелки, то будем считать, что знак угла положительный, в противоположном случае – отрицательный. Если точка лежит внутри многоугольника, то совершиться полный оборот вокруг исследуемой точки и сумма всех таких углов будет равна 2π или -2π, в зависимости от обхода многоугольника.

Недостатком метода является то, что при обходе всех вершин, во время вычисления угла возникает необходимость использования дорогостоящих операций, а именно обратных тригонометрических функций и квадратных корней.

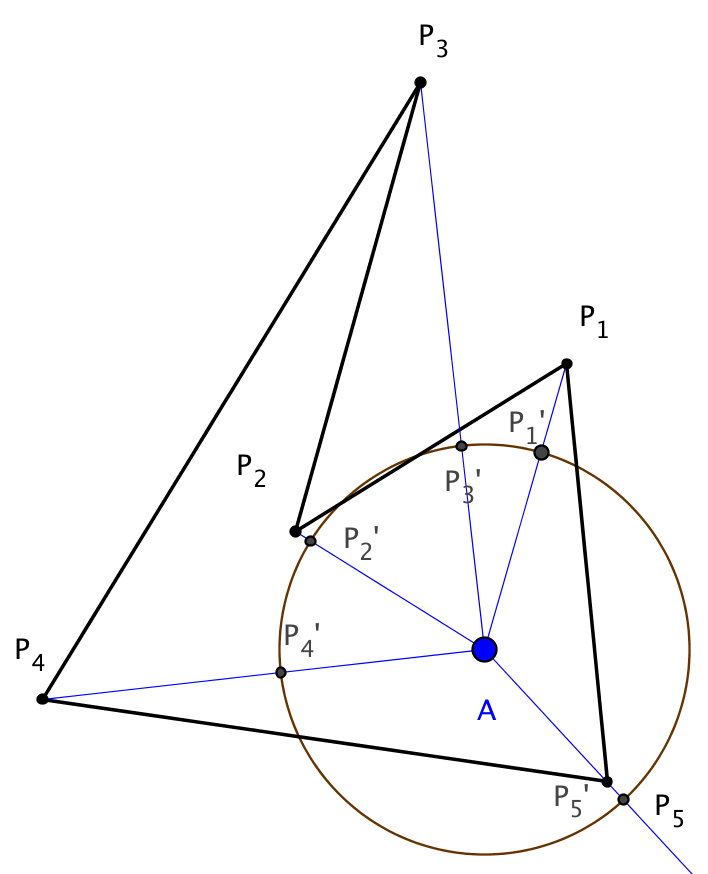


рис. 6. Визуализация метода суммирования углов

Этот метод можно использовать и для самопересекающихся многоугольников. Но в случае самопересекающихся многоугольников, поворот вокруг точки может быть не один, поэтому при принадлежности точки многоугольнику сумма углов будет равна 2πn или -2πn, где n – это количество оборотов, если сумма равна 0, то точка лежит снаружи многоугольника. Сложность алгоритма составляет O(N), где N – количество точек.

## **3.2. Реализация метода**

С помощью цикла, обойдем все стороны данного многоугольника. Определим угол через функцию arccos от аргумента, которым является отношение скалярного произведения к произведению длин векторов, образованных от вершин и исследуемой точки. Знак угла определим с помощью векторного произведения. Суммируем значение всех углов в цикле. После выхода из цикла сравниваем сумму с величиной 2π с установленной погрешностью. Если сумма равна 2π или -2π, то точка лежит внутри многоугольника, если числу близкому 0, то снаружи.

Преимуществом реализации в системе компьютерной алгебры «Wolfram Mathematica 13.1» является то, что существует функция суммирования Sum[], которая суммирует все углы и функция VectorAngle[], которая определяет значение угла между векторами.

1. **Метод №3: Ближняя точка и ее нормаль**

**4.1 Описание метода**

Суть алгоритма:

1. Для тестируемой точки ищем ближайшую точку на многоугольнике. При этом учитываем, что ближайшая точка может быть не только на отрезке, но и в вершине.
2. Ищем нормаль ближайшей точки. Если ближняя точка лежит на ребре, то нормаль это вектор, перпендикулярный ребру и смотрящий наружу многоугольника. Если ближняя точка - одна из вершин, то нормалью является усредненный вектор ребер, прилежащих к вершине.
3. Вычисляем угол между нормалью ближайшей точки и вектором от тестируемой точки до ближайшей. Если угол меньше 90 градусов, то мы - внутри, иначе - снаружи.

Причем угол как таковой считать не обязательно, достаточно проверить знак косинуса этого угла. Если положительный – внутри, если отрицательный – снаружи. А поскольку нас интересует только знак косинуса, то мы вычисляем знак скалярного произведения между двумя векторами.

Рассмотрим пример. Точка A1 (см. рисунке 7), ближайшая точка для нее находится на ребре. Нормаль к ребру параллельна вектору от тестируемой точки до ближайшей. В случае точки A1, угол между векторами = 0. Меньше 90 градусов, тестируемая точка A1 – внутри.

Протестируем точку A2. У нее ближайшая точка – вершина, нормаль к которой – усредненная нормаль ребер, прилегающих к этой вершине. Считаем скалярное произведение двух векторов, должно быть отрицательным. Мы – снаружи.

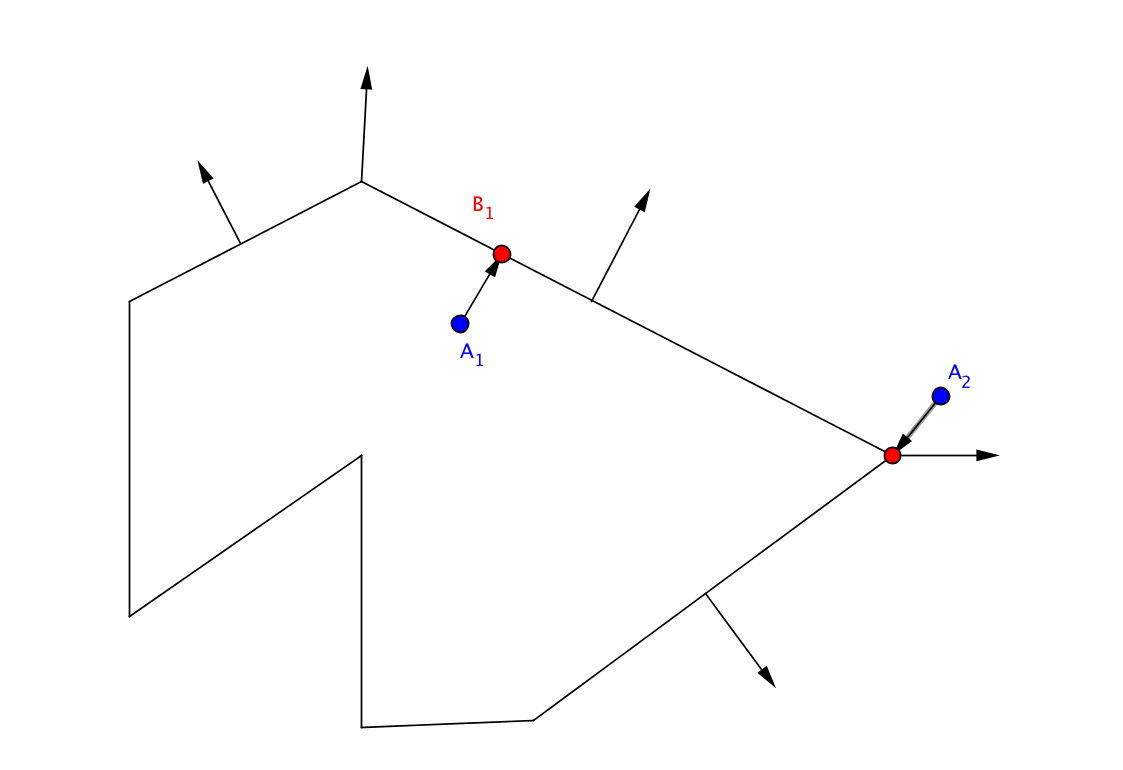


рис 7. Ближняя точка и ее нормаль

Алгоритмическая сложность данного алгоритма - O(log n). С вычислительной точки зрения метод, хотя и имеет подобную сложность, будет несколько помедленнее, чем трассировка. Прежде всего оттого, что расстояние между точкой и ребром чуть более дорогостоящая операция, чем пересечение двух линий. Неприятности, связанные с плавающей точкой, возникают в этом методе, как правило недалеко от ребер многоугольника. Причем чем мы ближе к ребру, тем больше вероятность неправильного определения знака.

К недостатку этого метода можно отнести прежде всего то, что этот метод требует, чтобы все нормали к ребрам были направлены в правильную сторону. То есть до того, как определять, снаружи мы или внутри, надо провести некую работу по вычислению этих нормалей и правильное их ориентирование. Если надо определить только для одной точки, процесс ориентации нормалей может занять неоправданно много времени.

**4.2 Реализация метода**

Сначала организуем цикл для обхода всех вершин, чтобы найти ту сторону многоугольника, которая содержит ближнюю точку. Нормаль к любой точке на этой стороне (кроме вершин) будет одинакова. Поэтому с помощью функции *met3Normal(V[i], V[i+1])* находим нормаль, где *V[i]­, V[i+1]* - это i-ая и i+1-ая вершина многоугольника. Также как и в предыдущем методе, с помощью функции arccos вычислим угол между векторами. Если он меньше 90 градусов, то точка внутри, если больше – снаружи.

1. **Сравнительная таблица методов**

|  |  |  |
| --- | --- | --- |
| Метод Трассировки лучей | + | * Простота |
| - | * Не применим в общем случае * Возникновение исключительных ситуаций при пересечении лучом искомой точки с вершиной многоугольника или ребро, которое частично совпадает с лучом. |
| Сложность | O(Log n) |
| Метод суммирования углов | + | * Самый стабильный метод |
| - | * Медленный * Не определяет точку, лежащую на ребре |
| Сложность | O(n) |
| Метод ближней точки и ее нормали | + | * Не возникает исключительных ситуаций, как в случае трассировки лучей |
| - | * Проблемы вычисления точки находящийся в непосредственной близости к полигону * Не выгоден для вычисления одной точки * Вычисление нормалей |
| Сложность | O(Log n) |

1. **Задача о принадлежности точки в произвольной плоскости**

Задачу о принадлежности точки многоугольнику также рассматривают в произвольной плоскости, формулировка задачи такая же, но добавляется к тому же z-ая координата для всех вершин и исследуемой точке.

Для решения этой задачи сначала необходимо проверить, принадлежит ли исследуемая точка плоскости данного многоугольника. Нужно спроецировать многоугольник на плоскость OXY и применить уже известные методы, то есть задача свелась к предыдущему пункту. Также нужно предусмотреть случай, когда проекцией многоугольника на OXY будет прямая. В этом случае исходный многоугольник необходимо спроецировать на плоскость OXZ и аналогично решить.

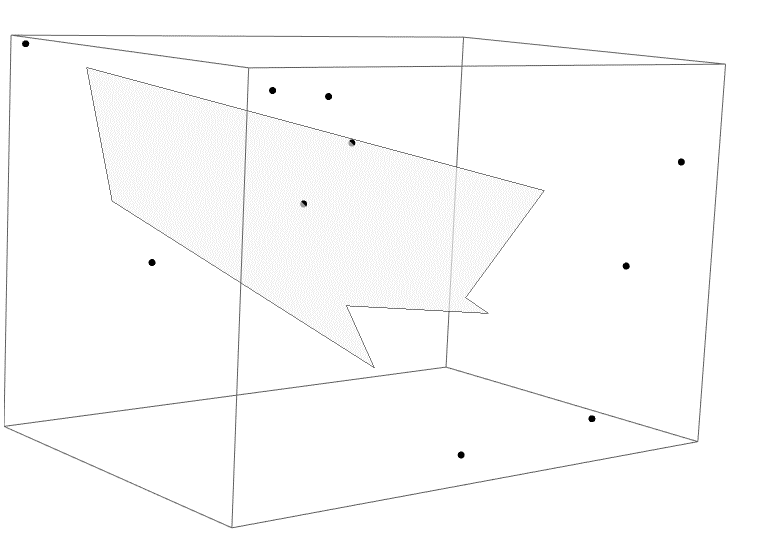


рис. 8. Многоугольник и точки в произвольной плоскости

# Заключение

В ходе выполнения задания ознакомительной практики была поставлена и решена задача о принадлежности точки многоугольнику. Рассмотрены различные методы решение задач и реализованы некоторые из них.

В процессе реализации данной задачи были изучены новые возможности языка программирования C++ и системы компьютерной алгебры «Wolfram Mathematica», а также закреплены знания, которые уже были получены ранее в рамках таких курсов как «Введение в специальность», «Введение в информационные технологии», а также «Информационные технологии профессиональной деятельности».

# Список литературы

1. Методы определения принадлежности точки многоугольнику // Хабр URL: <https://habr.com/ru/post/301102/>

(Дата обращения: 10.12.2022)

2. Задача о принадлежности точки многоугольнику // Википедия. Свободная энциклопедия.  
URL: https://ru.wikipedia.org/wiki/Задача\_о\_принадлежности\_точки  
(Дата обращения 10.12.2022)

3. Препарата Ф., Шеймос М.  // Вычислительная геометрия: Введение: Пер. с англ. — Москва: Мир, 1989.

4. Локализация точки в выпуклом многоугольнике // Хабр

URL: <https://habr.com/ru/post/144571/>

(Дата обращения: 10.12.2022)

5. Тестовое задание. Вхождение точки в произвольный полигон // Хабр  
URL: <https://habr.com/ru/post/283294>

(Дата обращения 10.12.2022)

6. Демонстрационный проект в Wolfram | Трассировка лучей  
URL: <https://demonstrations.wolfram.com/RayTracingForPointsInAPolygon>  
(Дата обращения 19.12.2022)