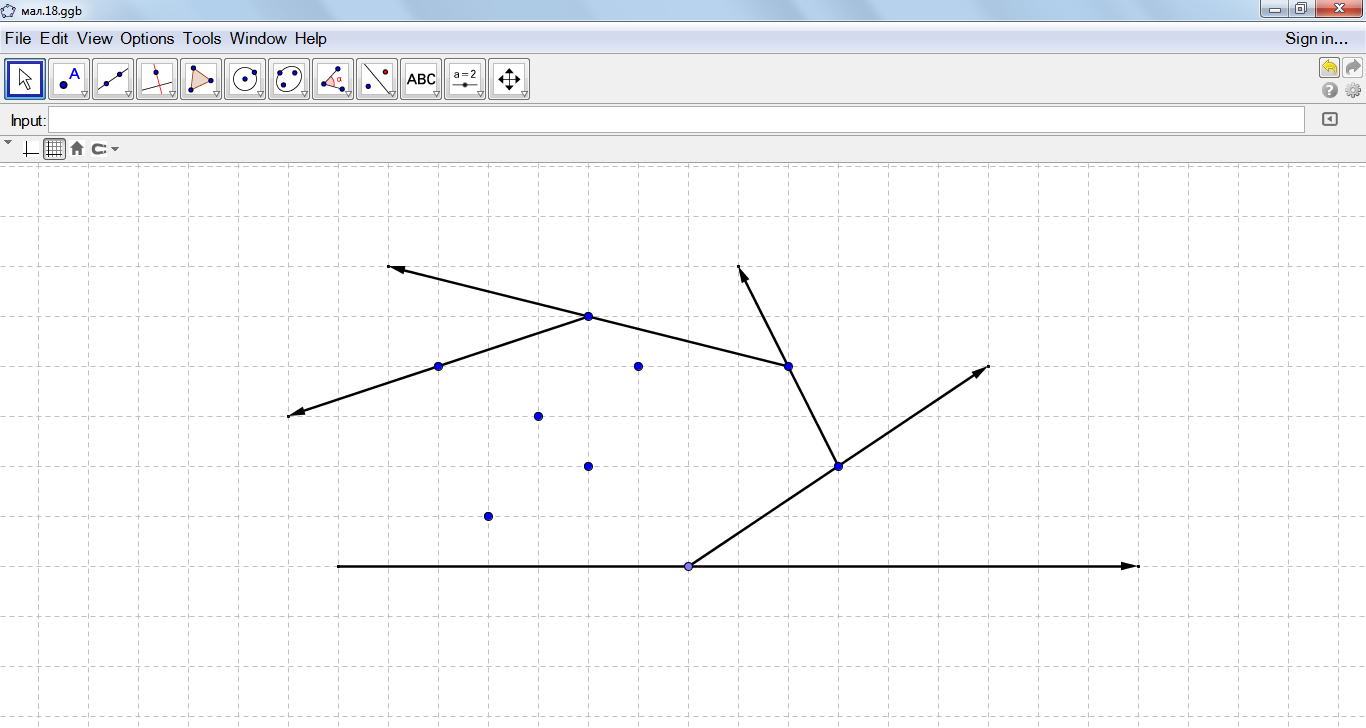
### Побудова опуклої оболонки для множини з N точок площини

Опуклою оболонкою деякої заданої безлічі точок називається перетин усіх опуклих великих кількостей, що містять задану множину. Для кінцевої безлічі точок опуклої оболонки завжди буде опуклий многокутник, усі вершини якого є точками початкової великої кількості.

Завдання полягає в тому, щоб для заданої кінцевої безлічі точок знайти вершини опуклої оболонки цієї великої кількості. Перераховуватимемо вершини в порядку обходу проти годинникової стрілки. Для ефективного вирішення цього завдання існує декілька різних алгоритмів. Приведемо найбільш просту реалізацію одного з них - алгоритму Джарвіса. Цей алгоритм іноді називають «завертанням подарунка».

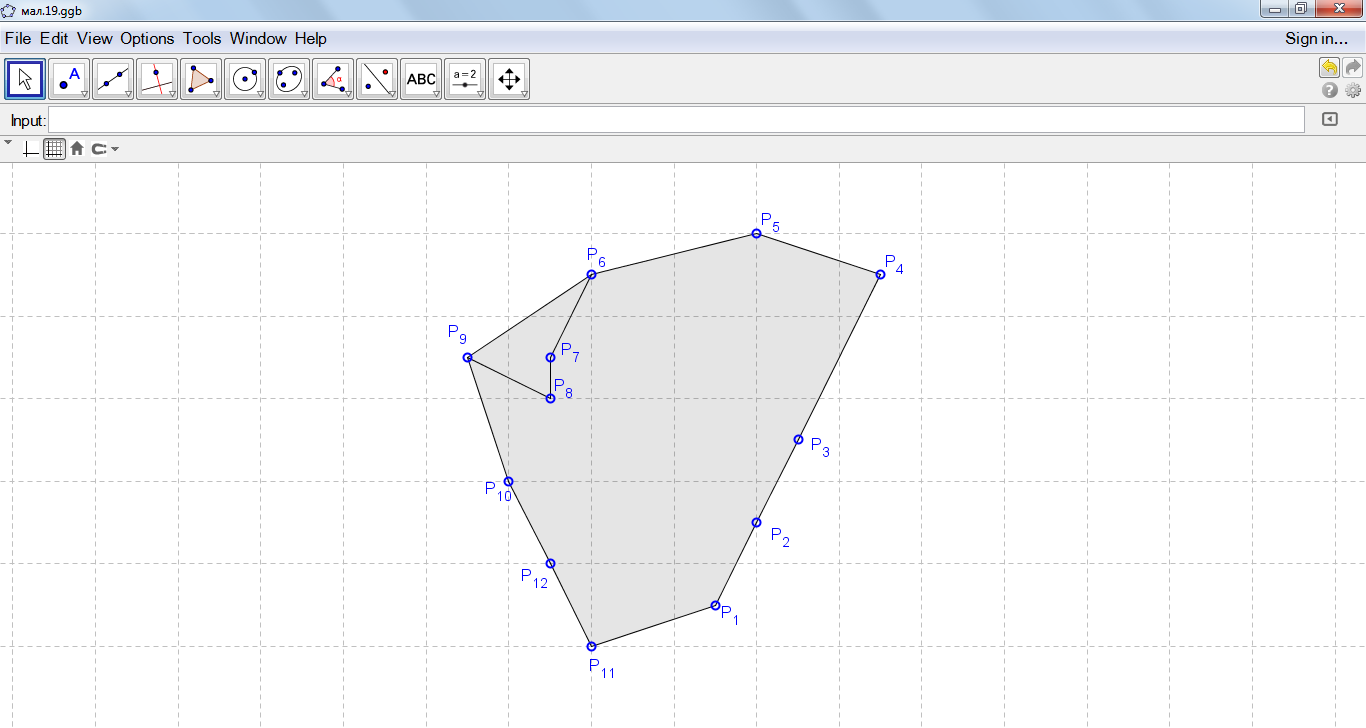


Перерахування точок шуканої межі опуклого многокутника розпочнемо з правої нижньої точки , яка свідомо належить межі опуклої оболонки. Позначимо її координати . Наступною при вказаному порядку обходу буде точка . Вона, очевидно, має ту властивість, що усі інші точки лежать «зліва» від вектору , тобто орієнтований кут між векторами і невід'ємний для будь-якої точки нашої множини. Для кандидата на роль точки перевіряємо виконання умови з усіма точками . Якщо точок, що задовольняють цій умові, декілька, то вершиною шуканого многокутника стане та з них, для якої довжина вектору максимальна.

Поступатимемо так само і далі. Припустимо, вже знайдена а вершина опуклої оболонки. Для наступної точки косі добутки невід’ємні для всіх точок . Якщо таких точок декілька, то вибираємо ту, для якої вектор має найбільшу довжину. Безпосередньо пошук такої точки можна здійснювати так. Спочатку ми можемо вважати наступною, , будь-яку точку. Потім обчислюємо значення , розглядаючи в якості усі інші точки. Якщо для однієї з них вказане вираження менше нуля, вважаємо наступною її і продовжуємо перевірку інших точок (аналогічно алгоритму пошуку мінімального елементу в масиві). Якщо ж значення вираження дорівнює нулю, то порівнюємо квадрати довжин векторів. В результаті за операцій чергова вершина опуклої оболонки буде знайдена. Продовжуючи цю процедуру, ми рано чи пізно повернемося до точки . Це означатиме, що опукла оболонка побудована.

При рішенні цієї задачі у разі спочатку цілочисельних координат ми повністю можемо уникнути застосування речової арифметики, а отже, позбавитися від втрати точності обчислень. Інакше в рішення можуть бути включені «зайві» точки, близькі до межі опуклої оболонки, або не враховані деякі з «потрібних» точок. Складність цього алгоритму складе , де – кількість точок в опуклій оболонці, у гіршому разі рівне .

Існує інший алгоритм рішення цієї задачі (алгоритм Грехема) з обчислювальною складністю , грунтований на попередньому сортуванні точок початкової великої кількості за значенням кута в полярній системі координат з центром в одній з точок опуклої оболонки. Тобто найбільш трудомістким завданням виявляється саме сортування вихідних точок. Сортування точок можна робити по знаку косого добутку , де – будь-яка вершина опуклої оболонки (наприклад, все та ж права нижня точка). У відсортованому масиві точок усі вказані твори мають бути невід'ємні. Точки з різними кутами () розташовуються в порядку збільшення довжин відповідних векторів .



Далі перегляд Грехема використовує стек, в якому зберігаються точки, що є кандидатами в опуклу оболонку. Спочатку в стек поміщається перша з відсортованих точок. Потім - сусідня з нею вершина опуклої оболонки. Якщо на першому з променів точок декілька, то це точка цього променя , найбільш віддалена від . Третя – точка . Нехай у вершині стоку знаходиться точка . Розглянемо наступну в порядку збільшення полярного кута точку початкової множини . Поки ділянкою ламаної не є опуклим, із стоку видаляється чергова точка . Потім поміщається в стік. У момент закінчення перегляду усіх точок в стоку знаходитимуться в точності усі вершини опуклої оболонки. Оскільки будь-яка точка додається в стік рівно один раз, то і віддаляється вона з нього не більше одного разу, тому час перегляду складає .