**Дніпропетровський ліцей інформаційних технологій  
при Дніпропетровському національному університеті  
імені Олеся Гончара**

**Випускна робота на тему:**

**Обчислювальна геометрія на площині**

**Виконавець:  
ліцеїст ІІІ-А-1 курсу  
Халін Олег Ігорович**

**Керівники роботи:**

**Сотніченко О. В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Боровик Л. І. \_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Дніпропетровськ  
2015**

# ЗМІСТ

[ВСТУП 4](#_Toc436577456)

[Тема роботи 4](#_Toc436577457)

[Мета роботи 4](#_Toc436577458)

[Актуальність роботи 4](#_Toc436577459)

[ОСНОВНА ЧАСТИНА 6](#_Toc436577460)

[Теоретична частина з предмету 6](#_Toc436577461)

[Побудова опуклої оболонки для множини з N точок площини 6](#_Toc436577462)

[Перевірка опуклості многокутника 9](#_Toc436577463)

[Перевірка приналежності точки внутрішньої області многокутника 9](#_Toc436577464)

[Коло, що «охоплює» N точок площини 11](#_Toc436577465)

[Теоретична частина з інформатики 12](#_Toc436577466)

[Елементи управління 12](#_Toc436577467)

[Опис роботи 13](#_Toc436577468)

[«Теорія» 13](#_Toc436577469)

[«Практика» 14](#_Toc436577470)

[Посібник користувача 14](#_Toc436577471)

[Панель елементів 14](#_Toc436577472)

[Панель задач 15](#_Toc436577473)

[Блок-схема 16](#_Toc436577474)

[Практичне застосування програми 16](#_Toc436577475)

[Програмно-апаратні вимоги 17](#_Toc436577476)

[Комплектація програми 17](#_Toc436577477)

[Використані програмні засоби 17](#_Toc436577478)

[ВИСНОВКИ 19](#_Toc436577479)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 20](#_Toc436577480)

[Література 20](#_Toc436577481)

[Інтернет-джерела 20](#_Toc436577482)

[ДОДАТКИ 21](#_Toc436577483)

[Додаток 1 (клас Dot) 21](#_Toc436577484)

[Додаток 2 (клас Polygon) 21](#_Toc436577485)

[Додаток 3 (перевірка опуклості) 22](#_Toc436577486)

[Додаток 4 (клас MyCircle) 22](#_Toc436577487)

[Додаток 5 (перевірка приналежності точки) 23](#_Toc436577488)

# ВСТУП

## Тема роботи

Тема роботи була обрана з декількох причин. По-перше, через зацікавленість у математиці та в точних науках взагалі. По-друге, через бажання зробити корисну програму, яка б стала в нагоді вчителям, які хочуть наочно продемонструвати засоби вирішення задач обчислювальної геометрії на площині, та учням, які вивчають тему самостійно та прагнуть її краще опанувати, також для олімпіадників при підготовці до олімпіад високого рівня. Врешті-решт була обрана дуже цікава, хоч і складна тема: «Обчислювальна геометрія на площині».

## Мета роботи

Основною метою роботи було дослідити методи вирішення задач обчислювальної геометрії та змоделювати деякі з них. Для досягнення цієї мети було поставлено наступні задачі:

1. Побудова опуклої оболонки для множини з N точок площини;
2. Перевірка опуклості многокутника;
3. Перевірка приналежності точки внутрішньої області многокутника;
4. Коло, що «охоплює» N точок площини.

## Актуальність роботи

Я вважаю, що тема «Обчислювальна геометрія на площині» є досить актуальною темою в наші дні, тому що:

* Ця тема є досить перспективною в програмуванні і має велику кількість алгоритмів які корисно запрограмувати, щоб в подальшому легше було зрозуміти дану тему;
* Для одинадцятикласників ця тема є новою, вона не вивчається в основному шкільному курсі, тож вона є не легкою в розумінні , чим не менш цікава;
* Переважна частина літератури з даного питання була випущена в останні 15-20 років і більшість з них на англійській мові, тож знайти насправді корисну інформацію для вирішення задач обчислювальної геометрії з метою їх подальшого кодування досить нелегко.

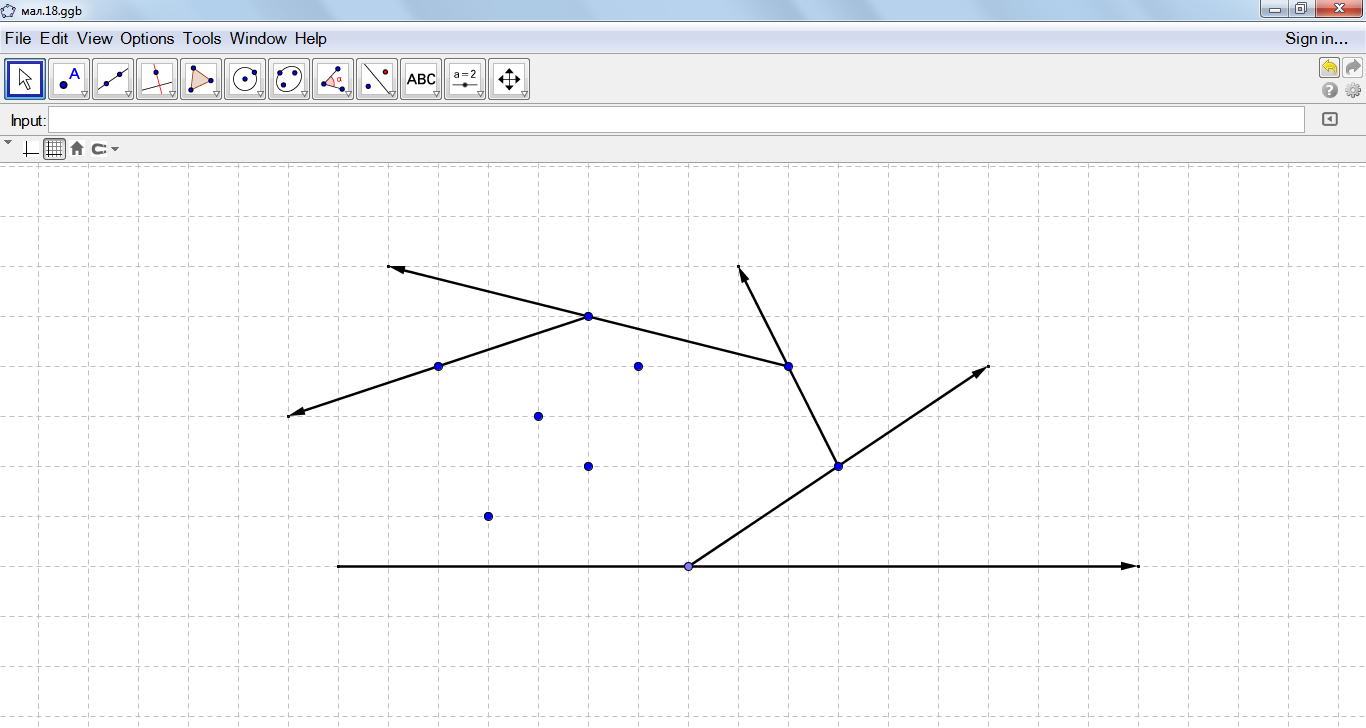
# ОСНОВНА ЧАСТИНА

## Теоретична частина з предмету

### 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».

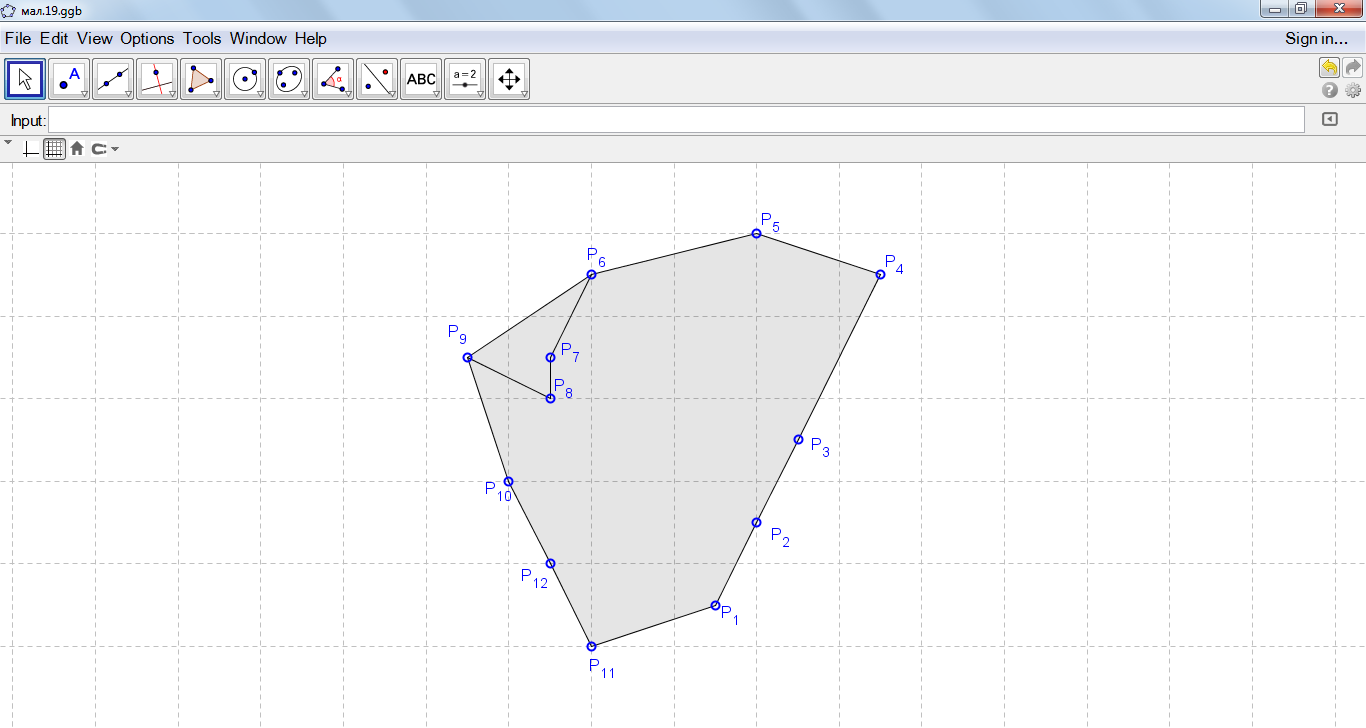


Перерахування точок шуканої межі опуклого многокутника розпочнемо з правої нижньої точки , яка свідомо належить межі опуклої оболонки. Позначимо її координати . Наступною при вказаному порядку обходу буде точка . Вона, очевидно, має ту властивість, що усі інші точки лежать «зліва» від вектору , тобто 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 .

### Перевірка опуклості многокутника

Опуклість многокутника з вершинами , , …, , перерахованими в порядку його обходу, легко перевірити, якщо вичислити знаки косих добутків , (тут є , a ). У опуклого многокутника знаки усіх вказаних добутків або недодатні, або невід'ємні(тобто знаки ненульових творів співпадають). Якщо ми знаємо напрям обходу, то знак косих добутків для опуклого многокутника визначений: при обході за годинниковою стрілкою усі косі добутки недодатні, а проти годинникової стрілки - невід'ємні.

### Перевірка приналежності точки внутрішньої області многокутника

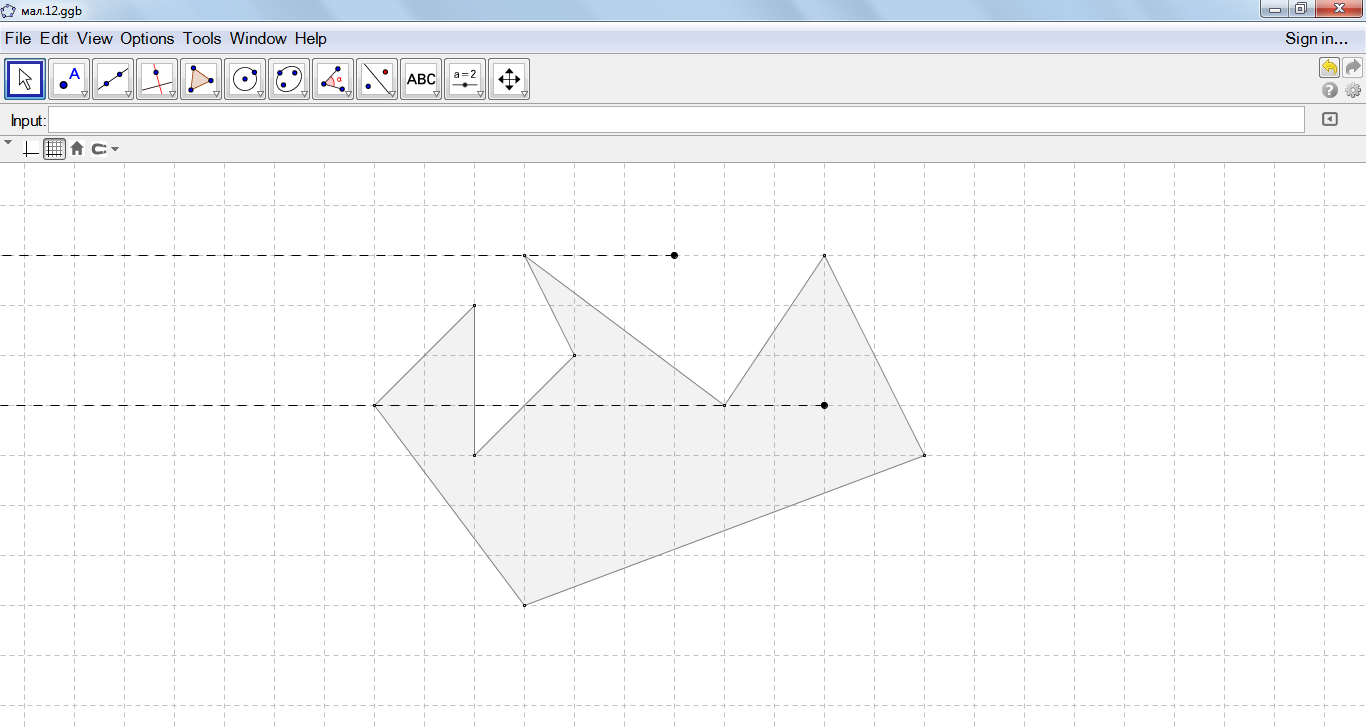
Нехай – деяка точка площини. Вимагається визначити її місцезнаходження відносно замкнутої ламаної, що є межею многокутника. Розглянемо спочатку випадок опуклого многокутника. Нехай задані своїми координатами вершини многокутника , , …, перераховані в порядку його обходу проти годинникової стрілки. При такому обході многокутник лежить ліворуч межі. І, значить, якщо точка лежить усередині многокутника, то орієнтований кут між векторами і негативний. Тому нам досить підрахувати величину косих добутків , ; значення береться по модулю . Якщо усі отримані при цьому значення від’ємні, то точка внутрішня. Якщо ж одне з них дорівнює нулю, а усі інші від’ємні, то належить межі многокутника (переконайтеся, що просто рівність нулю одного зі значень не досить). У протилежному випадку лежить поза нашим многокутником.

Розглянемо тепер довільний многокутник. Проведемо горизонтальний промінь з точки , наприклад, вліво. Оскільки многокутник обмежений, то завжди легко вказати на цьому промені точку , свідомо йому не прилеглу. Далі підрахуємо кількість перетинів відрізку з межею многокутника. Якщо воно дорівнює нулю або парне, то точка лежить поза многокутником, інакше - усередині нього.

Кількість перетинів відрізку з межею ми підрахуємо, розглянувши по черзі перетин відрізку з кожною з ланок ламаної. При цьому можливі наступні особливі випадки:

1. Одна з ланок ламаної прямої цілком міститься усередині відрізку .
2. Ланка ламаної торкається відрізку .
3. лежить на одній з ланок ламаної.

У останньому випадку належить межі многокутника, і в підрахунку загального числа перетинів необхідності немає. Для двох перших випадків поступимо таким чином. У першому випадку припинення ігноруватимемо. У другому - додатково перевіримо, "нижнім" або "верхнім" кінцем ланка ламаної торкається горизонтального відрізку . Якщо точкою дотику є "нижній" кінець ланки, то перетин ігнорується, а якщо "верхній" - то зачитується. З урахуванням цієї угоди торкання відрізку межі многокутника в одних точках ігнорується, а в інших точках вважається двічі. Це не змінить парності числа перетинів, а тільки вона важлива при пошуку відповіді на питання цього завдання. Якщо ж відрізок дійсно перетинає ламану в її вершині, то, за нашою угодою, число перетинів якраз збільшиться на одиницю (перетин з верхнім ребром зарахований не буде, а з нижнім - буде). Наприклад, на кількість перетинів для верхнього з досліджуваних точок буде рівна чотирьом (торкання зараховане двічі), а для нижньої точки - трьом (торкання не враховане, а перетин у вершині ламаній врахований один раз).



### Коло, що «охоплює» N точок площини

Це завдання полягає у відшукуванні координат центру кола мінімально можливого радіусу, усередині якої знаходяться усі задані точки. Іноді цю проблему називають мінімаксним завданням «про культурний центр». У ній вимагається по координатах будинків в місті підібрати місце для будівництва культурного центру так, щоб відстань до максимально віддаленого від нього будинку була мінімальною. Для того, щоб зрозуміти рішення цієї задачі в загальному випадку, розглянемо спочатку «трикутний» варіант: .

Навіть для трьох точок вид рішення залежить від їх взаємного розташування. Нехай точки лежать на одній прямій або утворюють тупокутний трикутник. Тоді шукана точка лежить на середині відрізку, що сполучає найбільш віддалені одна від однієї точки (в середині найбільшої сторони тупокутного трикутника). Насправді, відстань від цієї точки до будь-якої з перших двох зменшити не можна, а третя точка знаходиться на меншій відстані від знайденої точки, отже, вона лежить усередині кола, діаметр якого утворюють дві інші точки. А для гострокутного трикутника рішенням є центр описаного навколо нього кола (зміщення шуканої точки від нього у будь-якому напрямі приведе до збільшення відстані хоч би до однієї з точок). Прямокутний трикутник є «пограничним» для цих двох випадків, тобто для нього шукану точку можна знаходити будь-яким з описаних способів (звичайно, перший спосіб обчислювально більше за просту).

Для довільного N також є два випадки. Якщо знайдуться дві такі точки, що коло, побудоване на відрізку, що сполучає їх, як на діаметрі, містить усі інші точки (тобто для них виконується нерівність , де – центр кола), то це коло - шукане (фактично це випадок «тупокутного трикутника»). Якщо ж такої пари точок не знайшлося, то шукане коло свідомо проходить хоч би через три з вихідних точок. Тому тепер необхідно перебирати усі трійки точок до тих пір, поки не знайдеться така трійка, що коло, що проходить через ці точки, укладатиме усередині себе усі інші точки (випадок «гострокутного трикутника»).

## Теоретична частина з інформатики

Програма створена за допомогою мови програмування C#.

### Елементи управління

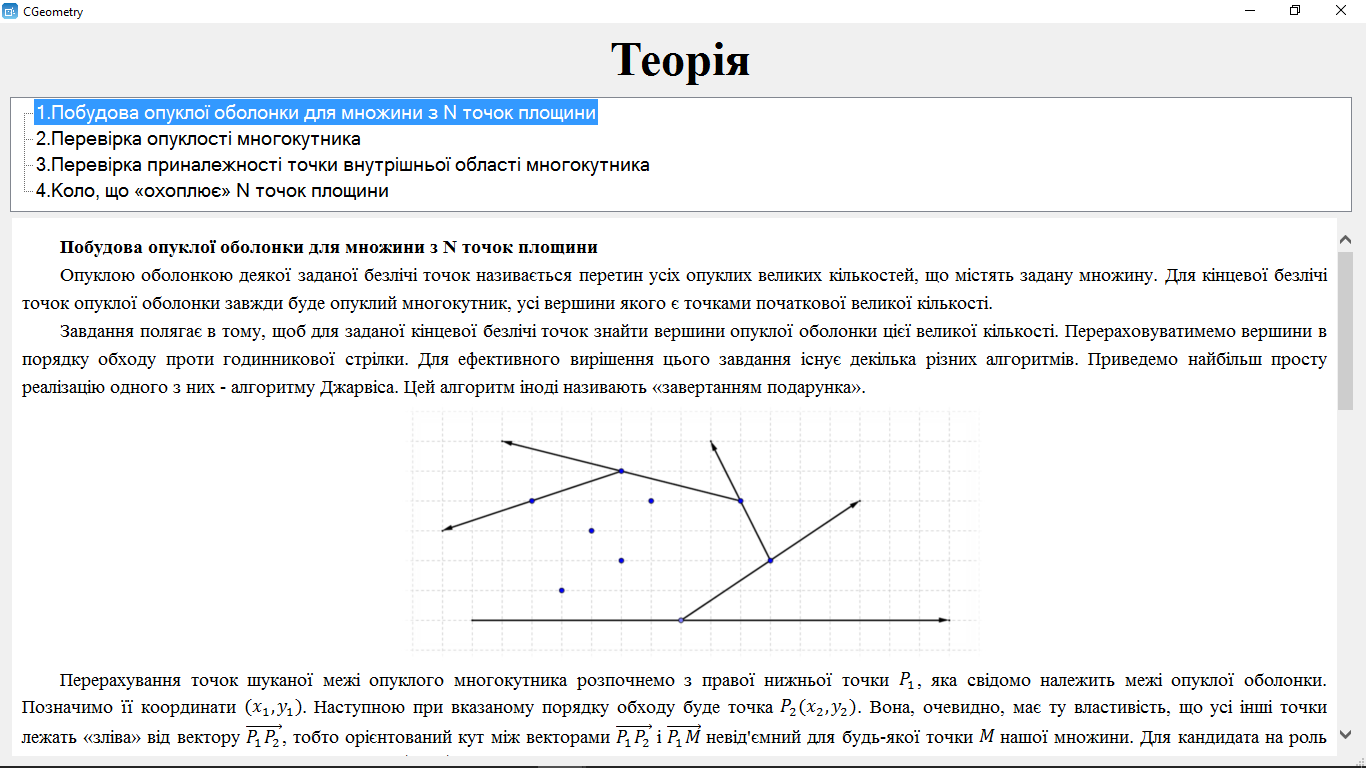
* *Window* – головне вікно, контейнер, в який поміщаються всі інші елементи управління;
* *ToolTip* – компонент, який відображає підказки при наведенні курсору миші на елементи керування;
* *PictureBox* – своєрідне полотно для виконання геометричних побудов;
* *GroopBox* - контейнер, у який можна помістити декілька компонентів;
* *Label* – елемент керування, який слугує для відображення тексту;
* *Button* – кнопка, при натисканні якої відбуваються деякі операції.
* *RichTextBox* – зчитує текстовий файл у форматі *.rtf;*
* *Click –* відбувається після кліку миші по елементу керування;
* *DoubleClick –* відбувається після подвійного кліку миші по елементу керування;
* *MouseDown/MouseUp –* відбувається при затисканій/відтисканій кнопці миші;
* *MouseMove –* відбувається при русі курсора по елементу керування.

## Опис роботи

Сайт: http://cgeometry.co.nf/.

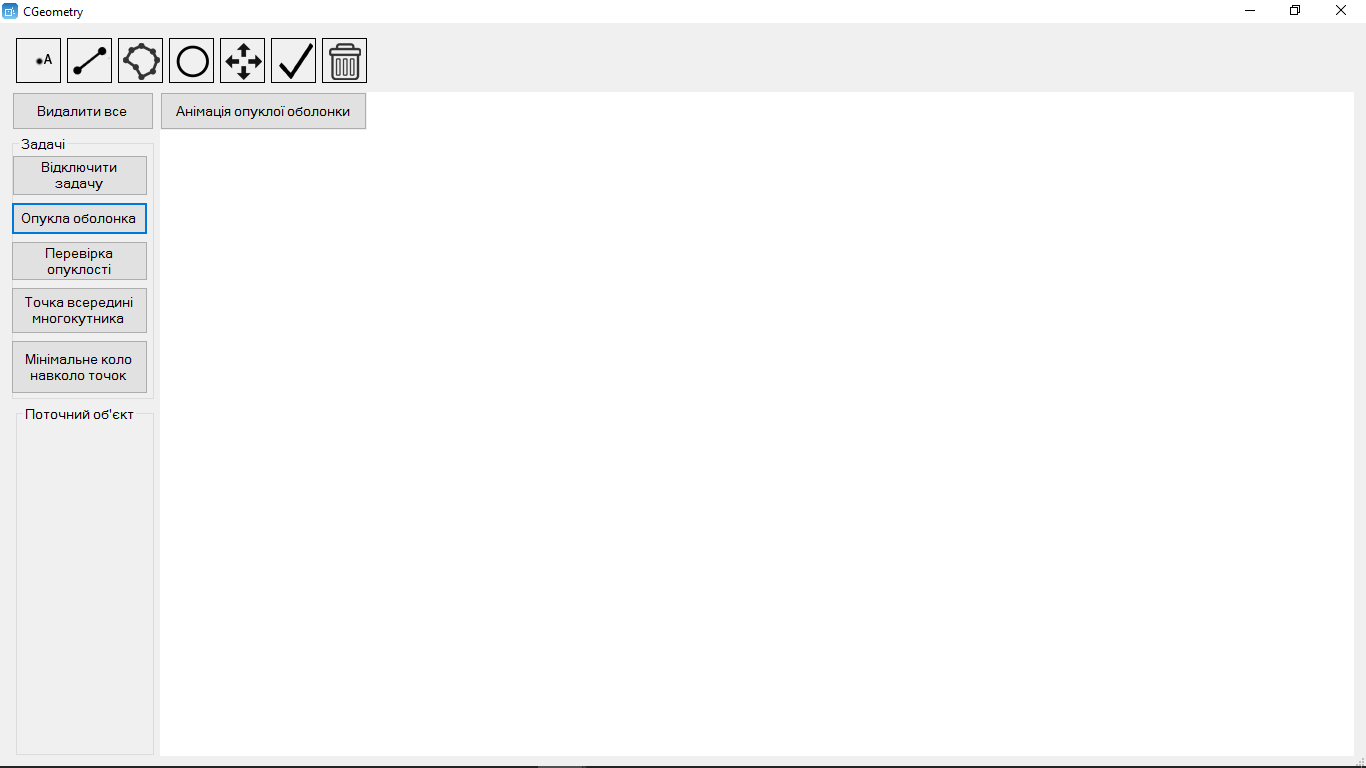
Сама робота складається з двох частин: «Теорія» та «Практика».

### «Теорія»



В частині «Теорія» знаходиться вся теорія з предмету яку я використовував при програмуванні алгоритмів модулювання задач обчислювальної геометрії.

### «Практика»



В частині «Практика» знаходяться елементарні побудови та змодульовані задачі обчислювальної геометрії.

## Посібник користувача

### Панель елементів

* Точка: Щоб побудувати точки необхідно в панелі елементів обрати режим "Точка" і розставити точки в потрібних місцях на полотні. Точки нумеруватимуться цілими числами;
* Відрізок: Щоб побудувати відрізок необхідно в панелі елементів обрати режим "Відрізок" і розставити дві точки в потрібних місцях на полотні. Точки нумеруватимуться літерами латинського алфавіту і цілими числами(A1, B1, ...). У панелі "Поточний об'єкт" буде вказуватися довжина відрізка;
* Многокутник: Щоб побудувати багатокутник потрібно в панелі елементів вибрати режим "Багатокутник" і розставити вершини багатокутника в потрібних місцях на полотні. Щоб завершити побудову багатокутника потрібно натиснути на кнопку "Завершити" в панелі елементів або двічі клікнути по полотну. Вершини багатокутника нумеруватимуться літерами латинського алфавіту і цілими числами(A1, B1, ...). У вікні "Поточний об'єкт" будуть вказуватися довжини сторін багатокутника і його опуклість;
* Коло: Щоб побудувати коло потрібно в панелі елементів вибрати режим "Коло" і розставити дві точки в потрібних місцях на полотні (перша - центр кола, друга - точка кола). У вікні "Поточний об'єкт" будуть вказуватися радіус кола і координати його центру;
* Перемістити: Щоб перемістити об'єкт потрібно в панелі елементів вибрати режим "Перемістити", навести курсор на об'єкт і перемістити його в потрібне місце, затиснувши кнопку миші;
* Закінчити: Щоб закінчити побудову об'єкта потрібно в панелі елементів натиснути кнопку "Закінчити";
* Видалити: Щоб видалити об'єкт потрібно в панелі елементів вибрати режим "Видалити", обрати об'єкт, який потрібно видалити і клікнути на нього.

### Панель задач

* Опукла оболонка: Щоб побудувати опуклу оболонку потрібно в панелі задач вибрати режим "Опукла оболонка" і розставити точки в потрібних місцях на полотні інструментом "Точка". Точки нумеруватимуться цілими числами. Оболонка буде будуватися автоматично. Після побудови опуклої оболонки слід в панеле завдань натиснути на кнопку "Відключити задачу";
* Перевірка опуклості: Щоб перевірити оболонку на опуклість потрібно в панелі задач вибрати режим "Перевірити на опуклість" і клікнути на багатокутник який потрібно перевірити на опуклість;
* Перевірка приналежності точки многокутнику: Щоб перевірити приналежність точки многокутнику потрібно в панелі задач вибрати режим "Точка всередині многокутника" і клікнути на місце розташування точки яку потрібно перевірити;
* Коло, що «охоплює» N точок площини: Щоб побудувати коло мінімального радіуса, що "охоплює" N точок площини потрібно в панелі задач обрати режим "Мінімальне коло навколо точок" і розставити точки в потрібних місцях на полотні. Коло будується автоматично. Після побудови кола слід в панеле завдань натиснути на кнопку "Відключити задачу".

## Блок-схема

## Практичне застосування програми

Дану програму можна використовувати:

* на уроках математики, для освоєння матеріалу;
* на уроках програмування;
* при самостійному освоєнні матеріалу;
* при підготовці до олімпіад з інформатики високого рівня;
* як посібник.

## Програмно-апаратні вимоги

Для повноцінної роботи програми необхідно задовольнити наступні вимоги:

Системні вимоги:

* NET Framework 4.5 або пізніше;
* Windows 8.1 або пізніші версії.

Апаратні вимоги:

* IntelCeleronDual-Coreprocessor або вище;
* Колонки (не обов’язково);
* монітор з роздільною здатністю 1366 х 768;
* 20 MB вільного місця на жорсткому диску.

## Комплектація програми

Щоб відкриту програму на своєму комп’ютері необхідно скачати собі папку «CGeometry» та запустити файл «CGeometry.exe»

1. CGeometry.exe – файл запуску програми.
2. Файли \*.jpg, \*.png, \*.gif, – це картинки, що використовуються програмою.
3. Файли \*.htm – це файли, що містять текстові дані для програми.
4. Документація.doc – файл з документацією до роботи;
5. ReadMe.txt – файл, необхідний для ознайомлення з програмою.

## Використані програмні засоби

* Visual Studio 2013-2015;
* Microsoft Word 2013;
* Gimp 2;
* Adobe Photoshop CS4
* Inkscape
* Paint.Net
* Sublime Text 3;
* Koala;
* Mozilla Firefox.

# ВИСНОВКИ

Задля виконання цієї роботи було досліджено відбудова 2-D графіки у мові програмування С#. Також, було змодельовано деякі задачі обчислювальної геометрії, було розглянуто їх особливості.

За час роботи над програмою я відкрив для себе багато нових аспектів програмування, а також вдосконалив раніше отримані знання як з програмування.

Також в процесі написання роботи я у великій мірі зрозумів, що основами сучасних прикладних наук є все ж таки фундаментальні науки, зокрема саме математичні знання є основою для програмістів. Створена мною програма дозволяє користувачу краще ознайомитися з темою «Обчислювальна геометрія на площині». Я сподіваюсь, що моя робота в цьому плані може бути корисною.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

## Література

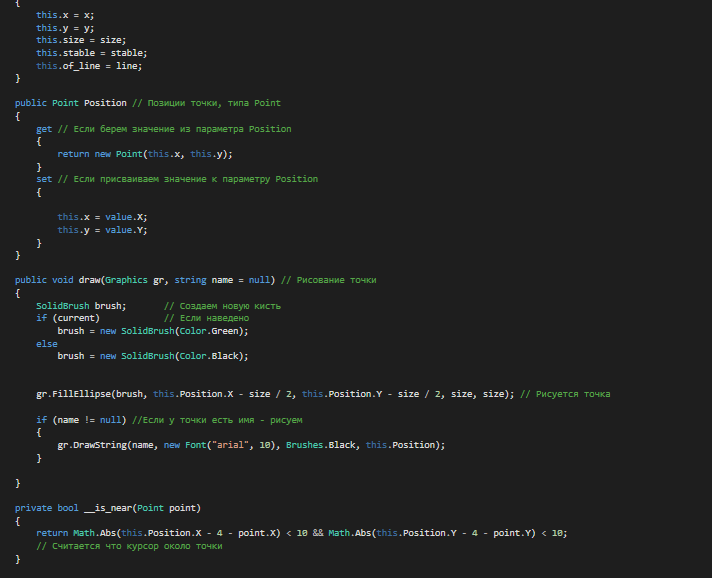
1. Кормен Т., Лейзерсон Ч., Ривести Р. «Алгоритмы. Построение и анализ». М.: МЦНМО, 2000.
2. Шикин Е.В., Боресков А.В., Зайцев А.А. «Начало компьютерной графики». М.: Диалог-МИФИ, 1993.
3. Курант Р., Роббинс Г. «Что такое математика». М.: ОГИЗ, государственное изд-во технико-теоретической литературы, 1947.
4. Окулов С.М. «100 задач по информатике». Киров: Изд-во ВГПУ, 2000.
5. Станкевич А.С. «Решение задач I Всероссийской командной олимпиады по программированию». Информатика № 12/2001.
6. Андреева Е.В. «Геометрические задачи на олимпиадах по информатике». Информатика № 14/2002.

## Інтернет-джерела

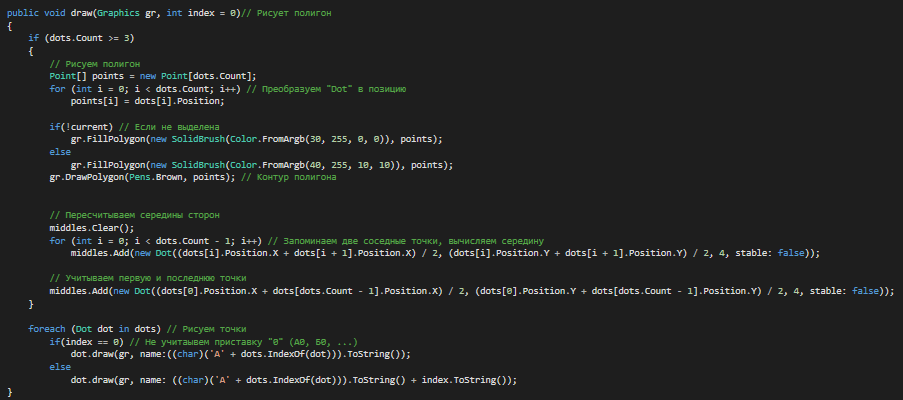
1. <http://msdn.microsoft.com/ru-ru/library>
2. <https://ru.wikipedia.org>
3. <http://habrahabr.ru/>
4. <http://www.cyberforum.ru/>
5. <http://stackoverflow.com/>
6. <http://getbootstrap.com/>
7. <http://daneden.github.io/animate.css/>
8. http://fortawesome.github.io/Font-Awesome/icons/

# ДОДАТКИ

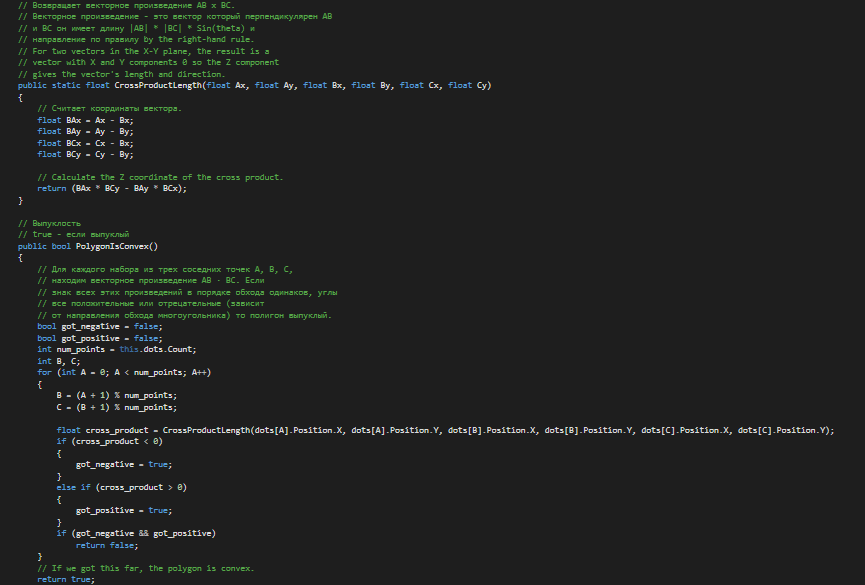
## Додаток 1 (клас Dot)



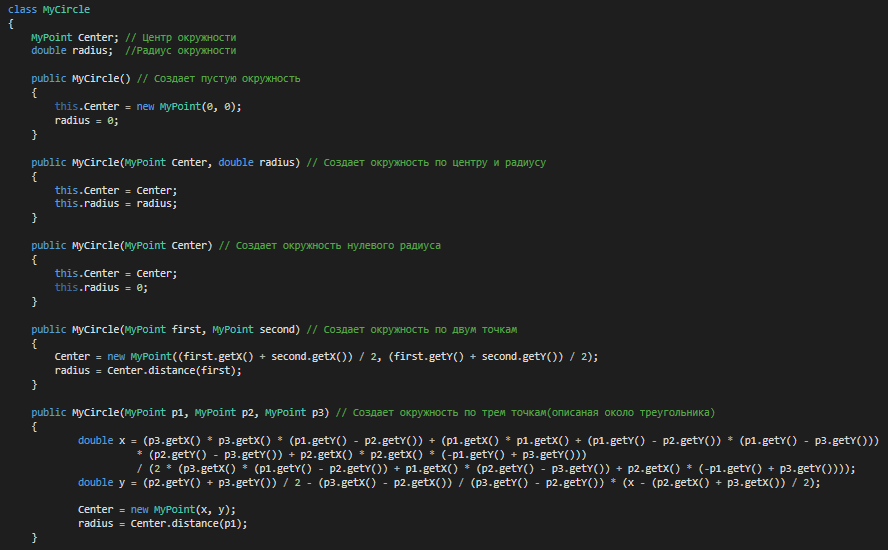
## Додаток 2 (клас Polygon)

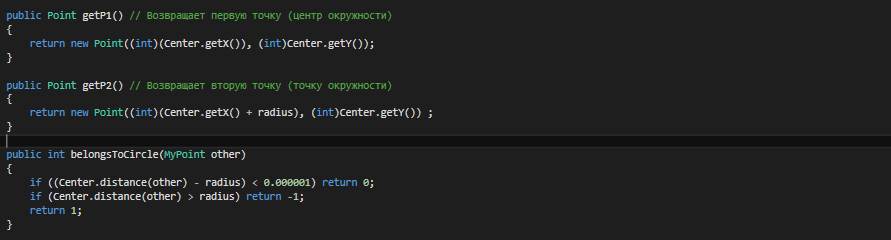


## Додаток 3 (перевірка опуклості)



## Додаток 4 (клас MyCircle)





## Додаток 5 (перевірка приналежності точки)

