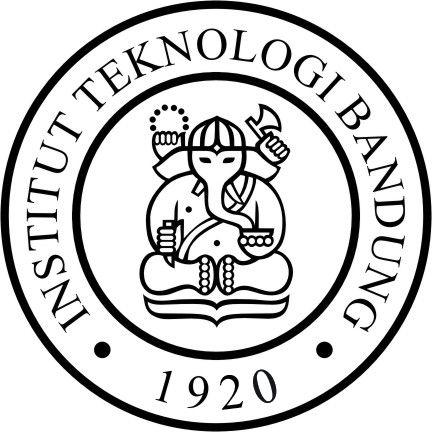
**LAPORAN TUGAS KECIL I IF-21**

**STRATEGI ALGORITMA**

**CONVEX HULL DENGAN ALGORITMA BRUTE FORCE**



|  |  |
| --- | --- |
| Disusun oleh : |  |
| Naufal Prima Yoriko | 13518146 |

**DAFTAR ISI**

|  |  |
| --- | --- |
| [**DAFTAR ISI**](#page2) | [**1**](#page2) |
| [**BAB I**](#page3) | [**3**](#page3) |
| [Garis](#page3) dan Persamaan Garis | [3](#page3) |
| [Convex](#page3) Hull | [3](#page3) |
| [Penjelasan](#page4) Langkah Algoritma | [3](#page4) |
| [Analisis](#page4) Kompleksitas Algoritma | [4](#page4) |
| [**BAB II**](#page6) | [**5**](#page6) |
| [Spesifikasi](#page6) Komputer | [5](#page6) |
| [Kode](#page6) Program | [5](#page6) |
| Uji Kasus | 9 |
| Ringkasan Hasil Uji Kasus | 12 |

|  |  |
| --- | --- |
| [**BAB III**](#page17) | [**13**](#page17) |
| [Kesimpulan](#page17) | [13](#page17) |
| [Catatan](#page19) Program | [13](#page19) |
| [**BAB IV**](#page23) | [**1**](#page23)**4** |

**BAB I**

**TEORI DASAR**

1. **Garis dan Persamaan Garis**

Di matematika, garis adalah objek dua dimensi yang merupakan kumpulan dari titik-titik.

Garis umumnya memiliki panjang tertentu, sehingga dapat diakatakan menghubungkan dua titik, misalnya P1 (x1,1, …. ,x1,n) dan P2 (x2,1, …. , x2,n), yang berada pada ujung-ujung dari garis tersebut.

Namun tidak jarang pula, garis dapat dinyatakan saja sebagai suatu persamaan pada ruang N-dimensi di matematika, sehingga tidak memiliki panjang. Salah satu persamaan umum dari garis di ruang 2 dimensi, dengan dua titik sembarang P1 (x1, y1) dan P2 (x2, y2), adalah

Dimana

Selain itu persamaan 1.1.1 dapat diubah nilai x dan y-nya dengan (x, y) suatu titik lain untuk menaksir posisi relatifnya terhadap garis tersebut.

**II. Convex Hull**

Dikutip dari spesifikasi tubes,

Himpunan titik pada bidang planar disebut convex jika untuk sembarang dua titik pada bidang tersebut (misal a dan b), seluruh segmen garis yang berakhir di a dan b berada pada himpunan tersebut.

Convex Hull dari himpunan titik S adalah himpunan convex terkecil yang mengandung S. Untuk dua titik, maka convex hull berupa garis yang menghubungkan 2 titik tersebut. Untuk tiga titik yang terletak pada satu garis, maka convex hull adalah sebuah garis yang menghubungkan dua titik terjauh. Sedangkan convex hull untuk tiga titik yang tidak terletak pada satu garis adalah sebuah segitiga yang menghubungkan ketiga titik tersebut. Untuk titik yang lebih banyak dan tidak terletak pada satu garis, maka convex hull berupa poligon convex dengan sisi berupa garis yang menghubungkan beberapa titik pada S.

1. **Penjelasan Langkah Algoritma**

Algoritma Convex Hull dengan Brute Force ini dapat di-*breakdown* langkahnya sebagai berikut

1. Program menerima input N pasang titik (x, y).
2. Program melakukan traversal pasangan P1 (x1, y1) dan P2 (x2, y2) dari seluruh titik yang di-input, kemudian dicari nilai a, b, dan c pada persamaan garis ax + by + c = 0.
3. Pada tiap langkah traversal di langkah kedua, melakukan iterasi seluruh titik lain P(X, Y) yang berjumlah N-2 dengan mensubtitusi persamaan garis ax + by + c = 0 menjadi f(X, Y) = aX + bY + c.
4. Apabila garis tersebut adalah garis terluar (*borderline*), yang dibuktikan dengan seluruh nilai hasil subtitusi titik lain pada persamaan memiliki tanda yang sama (+, -, atau 0), maka titik penyusun P1 (x1, y1) dan P2 (x2, y2) merupakan bagian dari himpunan titik Convex Hull.

1. **Analisis Kompleksitas Algoritma**

Algoritma Convex Hull di atas dapat dijabarkan kompleksitas masing-masingnya sebagai berikut

1. Traversal pasangan titik sejumlah N akan melakukan traversal sebanyak jumlah pasangan, atau N(N-1)/2 pasang titik, dengan kata lain kompleksitasnya O(n2).
2. Tiap traversal menghitung persamaan garis ax + bx + c = 0, yang mana kompleksitasnya O(1) atau konstan.
3. Di traversal yang sama, mengiterasi nilai f(X, Y) = aX + bY + c dari N-2 titik, menghitung persamaan kompleksitasnya O(1) dan dilakukan N-2 kali, sehingga keseluruhan langkah ketiga ini komplesitasnya O(n)

Berdasar langkah tersebut dapat kita ketahui kompleksitas akhir algoritma ini yakni O(n3), yang berasal dari O(n2) \* O(1) \* O(n).

**BAB II**

**IMPLEMENTASI**

1. **Spesifikasi Komputer**

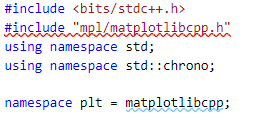
Pada pengerjaan tugas kecil ini saya menggunakan sebuah laptop dengan spesifikasi sebagai berikut

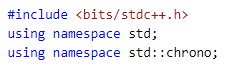
|  |  |
| --- | --- |
| Merk dan Model Laptop | HP Pavilion Laptop 14-bf1xx |
| BIOS | F.34 |
| Processor | Inter Core i7-8550U CPU @1.80 GHz (3 CPUs) |
| Memory RAM | 3300 MB |
| DirectX Version | DirectX 12 |
| Display GPU | Intel UHD Graphics 620 (64 MB) |
| Render GPU | NVIDIA GeForce 940MX (4055 MB) |
| OS | Kubuntu 18.04 (Virtual Machine) |

Tabel 2.1. Tabel Spesifikasi Laptop

1. **Kode Program**

Berikut ini adalah program yang saya tulis dalam bahasa C++ dan akan saya pisah penjabarannya per fungsi dan struktur agar mudah untuk dipahami fungsionalitasnya

1. Header (Library, Namespace, dll)

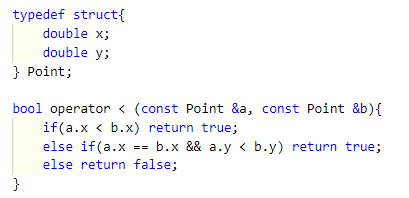


Gambar 2.2.1.1 Header ch.cpp Gambar 2.2.1.2 Header ch\_nolinux.cpp

*Sumber : Dokumen Pribadi Sumber : Dokumen Pribadi*

Header ini menunjukkan library dan namespace apa saja yang digunakan untuk menunjang program ini. Pada gambar 2.2.1.1 header tersebut juga memuat library matplotlibcpp yang berfungsi untuk membuat gambar grafik dari Convex Hull, namun sayangnya hanya bisa dijalankan di linux, sehingga ada versi program yang lebih sederhana, tanpa matplotlibcpp dengan header seperti ditunjukkan gambar 2.2.1.2.

1. Struktur Point

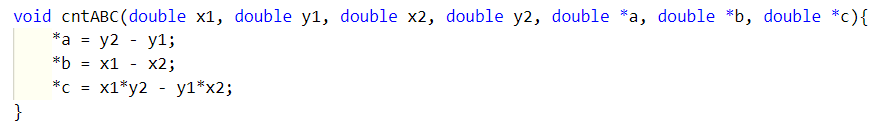


Gambar 2.2.2 Struktur Point dan definisi operator <

*Sumber : Dokumen Pribadi*

Struktur Point pada gambar 2.2.2 berfungsi untuk membungkus dua nilai double menjadi satu kesatuan yang mana sangat berguna untuk merepresantasikan koordinat titik 2 dimensi yang ada. Sedangkan operator ‘<’ didefinikan ulang pada struktur ini dengan fungsi di bawahnya.

1. Fungsi cntABC

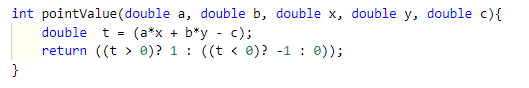


Gambar 2.2.3 Fungsi cntABC

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.3 terdapat fungsi cntABC yang berfungsi untuk menentukan nilai a,b, dan c dari persamaan garis ax + by = c dari dua titik.

1. Fungsi pointValue

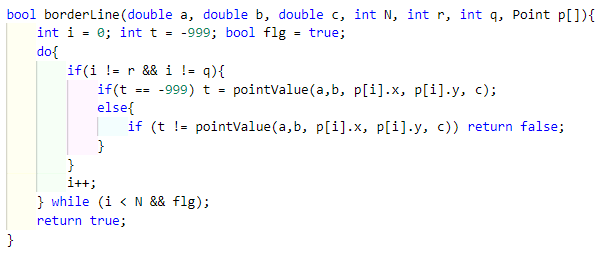


Gambar 2.2.4 Fungsi lineValue

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.4 terdapat fungsi lineValue yang berfungsi untuk menentukan posisi relatif suatu titik terhadap sebuah garis, dan mendiskritkan posisinya menjadi tiga kemungkinan hasil {-1, 0, 1} tergantung dari posisinya.

1. Fungsi borderline

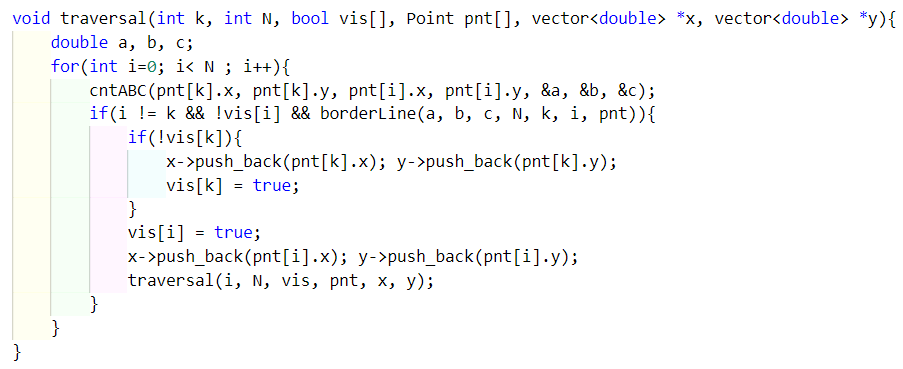


Gambar 2.2.5 Fungsi borderLine

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.5 terdapat fungsi borderLine yang berfungsi untuk menentukan apakah suatu garis (*line*) dari dua buah titik merupakan pembatas (*border*) / sisi luar dari bidang Convex Hull yang akan dibuat.

1. Fungsi traversal

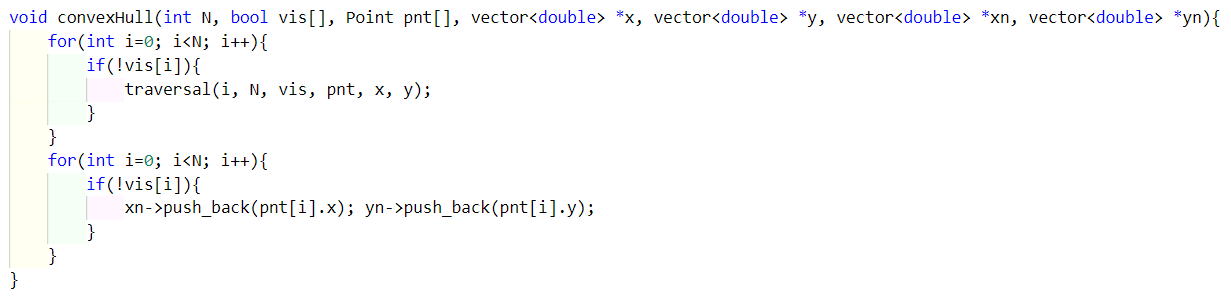


Gambar 2.2.6 Fungsi traversal

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.6 terdapat fungsi traversal yang berfungsi untuk melakukan traversal/penelusuran guna mencari pasangan dua buah titik untuk diuji apakah garis yang dihasilkan termasuk sisi terluar (*borderline*) dari Convex Hull atau tidak.

1. Fungsi convexHull

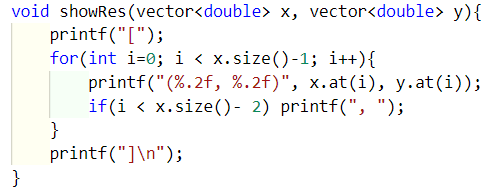


Gambar 2.2.7 Fungsi convexHull

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.7 terdapat fungsi convexHull yang berfungsi untuk melakukan inisiasi dari fungsi traversal/penelusuran dan juga menuliskan titik yang bukan termasuk himpunan penyusun sisi terluar (*borderline*) ke dalam suatu kontainer.

1. Fungsi showRes

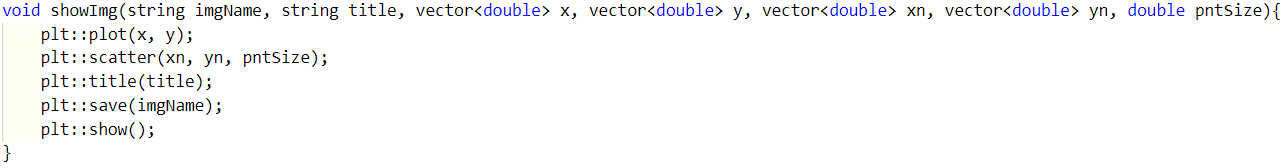


Gambar 2.2.8 Fungsi showRes

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.8 terdapat fungsi showRes yang berfungsi untuk menampilkan hasil berupa himpunan titik penyusun sisi terluar (*borderline*) ke layar/konsol (ataupun file, sesuai pengaturan di fungsi CHProg).

1. Fungsi showImg

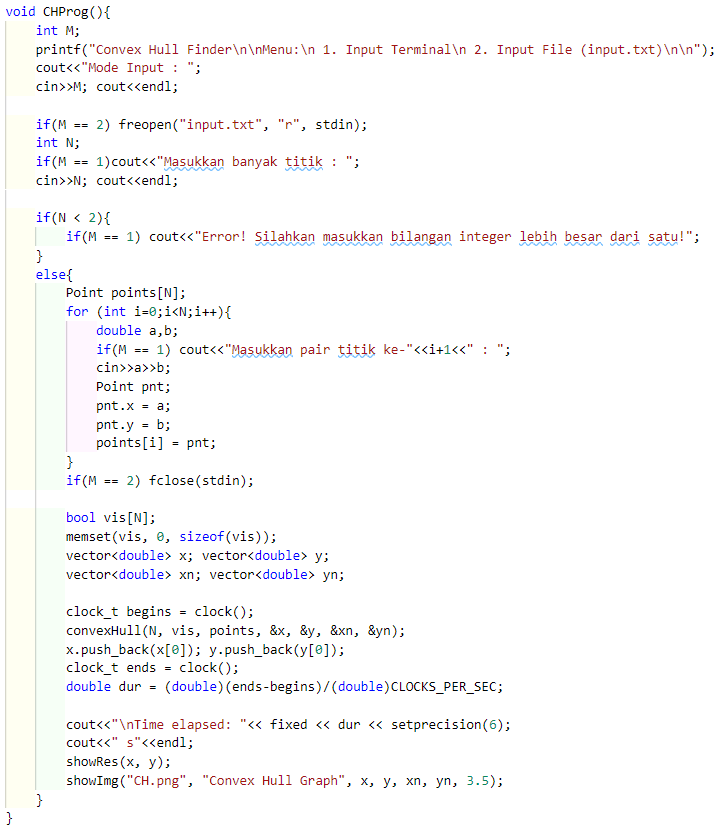


Gambar 2.2.9 Fungsi showImg

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.9 terdapat fungsi showImg yang berfungsi untuk menampilkan gambar grafik dari Convex Hull yang sudah dihasilkan ke layer sekaligus menyimpannya sebagai file ‘CH.png’. Sebagai catatan, karena fungsi ini membutuhkan library matplotlibcpp, maka fungsi ini hanya ada di program ch.cpp yang mana hanya dapat dijalankan di linux saja, sedangkan pada ch\_nolinux.cpp tidak ada.

1. Fungsi CHProg

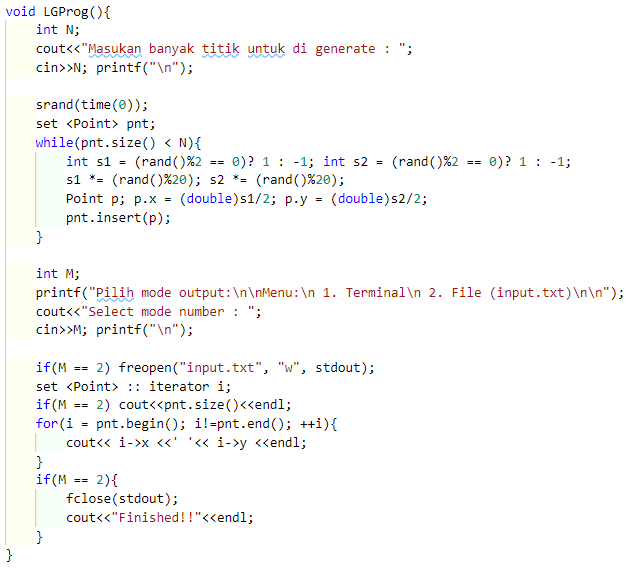


Gambar 2.2.10 Fungsi CHProg

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.10 terdapat fungsi CHProg yang merupakan salah satu dari dua subprogram utama yang berfungsi untuk mencari Convex Hull dari himpunan titik-titik yang diberikan pengguna (ditampilkan dalam bentuk gambar grafik dan teks di konsol) sekaligus menampilkan waktu/durasi eksekusi dari fungsi convexHull. Titik yang diberikan dapat dituliskan secara langsung di konsol maupun dibaca dari file ‘input.txt’ yang telah dituliskan sebelumnya.

1. Fungsi LGProg

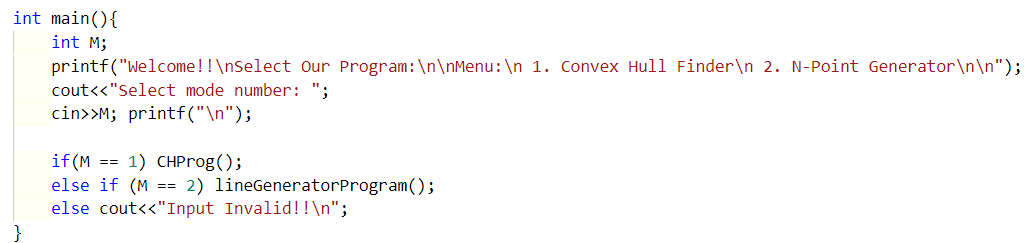


Gambar 2.2.11 Fungsi LGProg

*Sumber : Dokumen Pribadi*

Pada gambar 2.2.11 terdapat fungsi LGProg yang merupakan salah satu dari dua subprogram utama yang berfungsi untuk membuat himpunan titik-titik yang berbeda secara acak sejumlah yang diminta user. Hasil dapat dibaca secara langsung dari konsol maupun dituliskan ke file ‘input.txt’.

1. Main Function / Program Utama



Gambar 2.2.12 Main Function / Program Utama

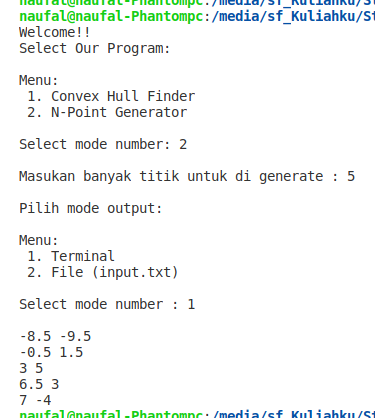
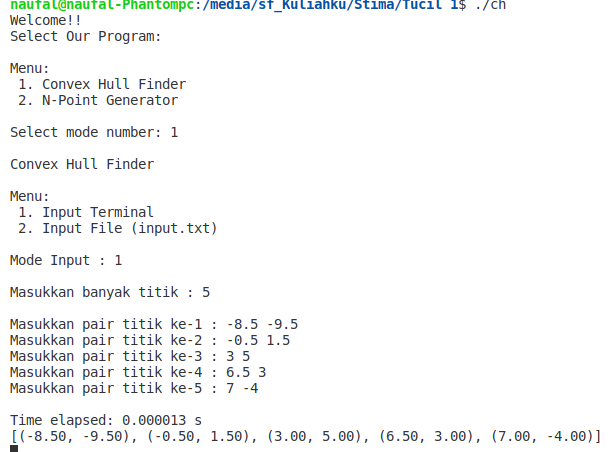
*Sumber : Dokumen Pribadi*

Pada gambar 2.2.12 berisi *main function* atau program utama yang berfungsi untuk memilih subprogram mana yang ini dijalankan, antara CHProg untuk mencari Convex Hull atau LGProg untuk men-*generate* titik acak dengan jumlah tertentu.

1. **Uji Kasus**

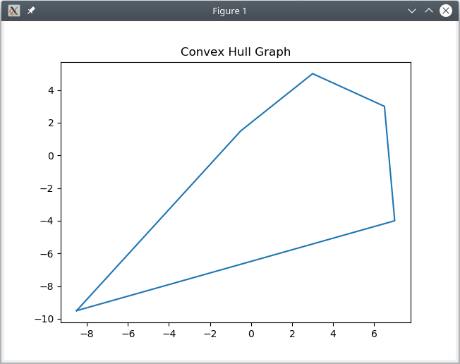
Berikut akan saya berikan contoh input dan output program ini pada beberapa kasus uji

1. Uji kasus **N = 5**



Gambar 2.3.1.1 Generate Titik Uji N = 5 Gambar 2.3.1.2 Hasil Convex Hull Titik N = 5

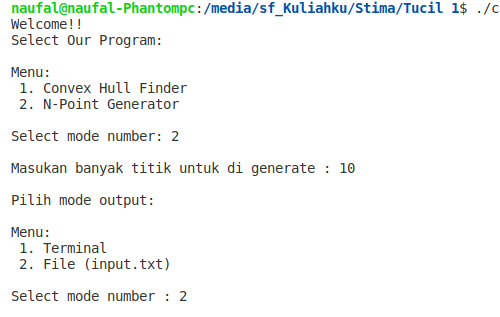
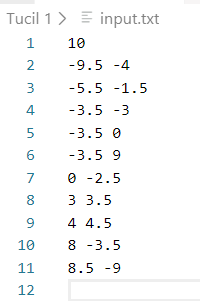
*Sumber : Dokumen Pribadi Sumber : Dokumen Pribadi*



Gambar 2.3.1.3 Grafik Hasil Convex Hull Titik N = 5

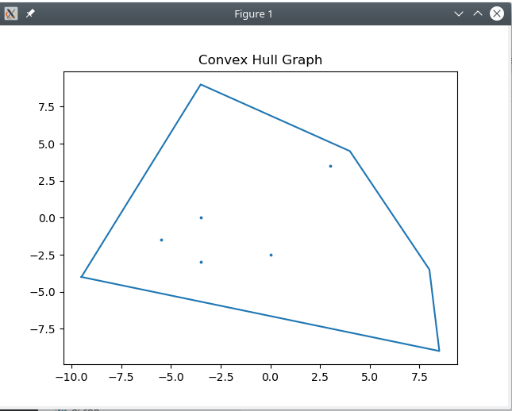
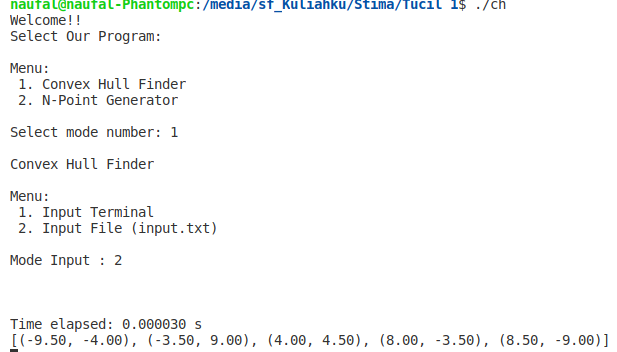
*Sumber : Dokumen Pribadi*

1. Uji kasus **N = 10**



Gambar 2.3.2.1 Generate Titik Uji N = 10 Gambar 2.3.2.2 Data input.txt Titik N = 10

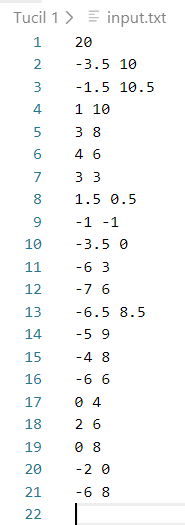
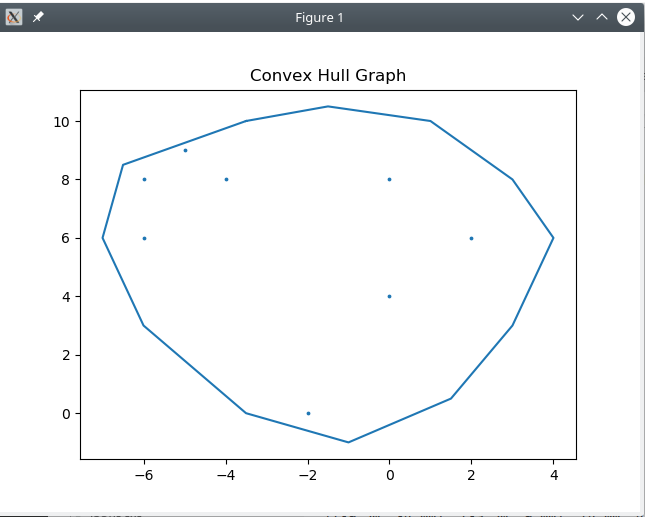
*Sumber : Dokumen Pribadi Sumber : Dokumen Pribadi*



Gambar 2.3.2.3 Hasil Convex Hull Titik N = 10 Gambar 2.3.2.4 Grafik Hasil Convex Hull Titik N = 10

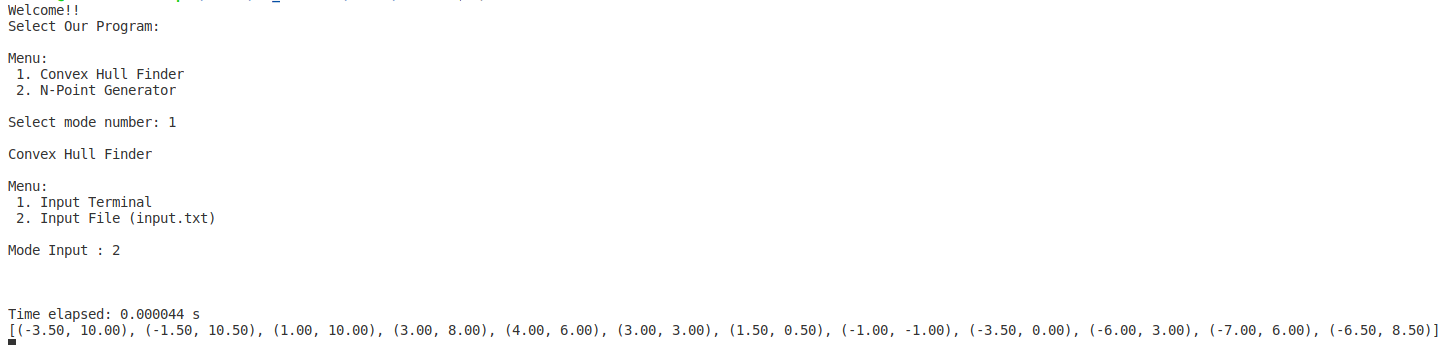
*Sumber : Dokumen Pribadi Sumber : Dokumen Pribadi*

1. Uji kasus **N = 20**



Gambar 2.3.3.1 Data input.txt Titik N = 20 Gambar 2.3.3.2 Grafik Hasil Convex Hull Titik N = 20

*Sumber : Dokumen Pribadi Sumber : Dokumen Pribadi*



Gambar 2.3.3.3 Hasil Convex Hull Titik N = 20

*Sumber : Dokumen Pribadi*

**IV. Ringkasan Hasil Uji Kasus**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **N** | **Titik Uji** | | **Titik Convex Hull** | | **Waktu Kompilasi (s)** | **Kebenaran** |
| **x** | **y** | **x** | **y** |
| 5 | -8.5  -0.5  3  6.5  7 | -9.5  1.5  5  3  -4 | -8.5  -0.5  3  6.5  7 | -9.5  1.5  5  3  -4 | 0.000013 | YA |
| 10 | -9.5  -5.5  -3.5  -3.5  -3.5  0  3  4  8 8.5 | -4  -1.5  -3  0  9  -2.5  3.5  4.5  -3.5  -9 | -9.5  -3.5  4  8  8.5 | -4  9  4.5  -3.5  -9 | 0.000030 | YA |
| 20 | -3.5  -1.5  1  3  4  3  1.5  -1  -3.5  -6  -7  -6.5  -5  -4  -6  0  2  0  -2  -6 | 10  10.5  10  8  6  3  0.5  -1  0  3  6  8.5  9  8  6  4  6  8  0  8 | -3.5  -1.5  1  3  4  3  1.5  -1  -3.5  -6  -7  -6.5 | 10  10.5  10  8  6  3  0.5  -1  0  3  6  8.5 | 0.000044 | YA |

Tabel 2.4. Tabel Ringkasan Hasil Uji Kasus

**BAB III**

**PENUTUP**

1. **Kesimpulan**

Berdasarkan hasil implementasi di bab dua, dapat disimpulkan hasil dari program sebagai berikut

|  |  |  |  |
| --- | --- | --- | --- |
| No | Poin | Ya | Tidak |
| 1 | Program berhasil dikompilasi | v |  |
| 2 | Program berhasil *running* | v |  |
| 3 | Program dapat menerima input dan menuliskan output | v |  |
| 4 | Hasil sudah benar untuk semua n | v |  |
| 5 | Program dapat menampilkan gambar grafik dari hasil Convex Hull (Bonus) | v |  |

1. **Catatan Program**

Program memang berjalan sangat baik dan sesuai degan yang diinginkan, namun tetap memerlukan untuk memerhatikan beberapa hal diantaranya

1. Library matplotlibcpp yang saya gunakan sejatinya adalah library matplotlib python sehingga diperlukan linking ke library python saat kompilasi, yang mana command untuk compile (di Kubuntu 18.04, kemungkinan sama untuk system operasi linux turunan Debian lainnya)

g++ ch.cpp -o ch -I/usr/include/python2.7 -lpython2.7

Source code untuk versi yang harus di Linux ini saya beri nama ch.cpp

1. Matplotlibcpp karena terhubung dengan library di python maka untuk dijalankan memerlukan beberapa library python lainnya, beberapa yang saya tahu dibutuhkan : numpy, matplotlib, libpython2.7-dev . Namun tidak menutup kemungkinan membutuhkan library tambahan lain.
2. Untuk linking pada sistem operasi selain ubuntu/linux, misal windows mengalami beberapa kendala,sehingga tidak bisa dikompilasi, maka saya membuat file src khusus di windows ch\_nolinux.cpp yang mana bisa dikompilasi dengan biasa (tanpa perlu linking), namun karena tidak ada library matplotlibcpp maka berarti di versi ini tidak ada penampilan gambar.

**BAB IV**

**REFERENSI**

Levitin, Duc Anany V. *Lecture 03 Brute Force*. Diakses pada 17-21 Januari 2020, dari http://mslab.csie.asia.edu.tw/~ktduc/AA/Lecs/Chapter03%20Duc%20Anany%20V.%20Levitin%203e.pdf/

Anonim. *Penyelesaian Persoalan Convex Hull dengan Algoritma Brute Force*. Diakses pada 19-21 Januari 2020, dari http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/Tugas-Kecil-1-(2020).pdf/

Anonim. *Computational Geometry : Convex Hull*. Diakses pada 21 Januari 2020, dari http://www.cs.jhu.edu/~misha/Fall05/09.13.05.pdf/

Anonim. *Brute Force Algorithms CS 351, Chapter 3*. Diakses pada 21 Januari 2020, dari http://math.uaa.alaska.edu/~afkjm/cs351/handouts/bruteforce/