**Laporan Tugas Kecil I**

IF2211 – Strategi Algoritma

Convex HuLL

Disusun oleh

Stephen Thajeb – 13518150 – K3



**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2019**

1. **Algoritma Brute Force dan Kompleksitas**

Algoritma Brute Forces dan Kompleksitas untuk Mencari Himpunan Titik Pembentuk Convex Hull sebagai berikut :

1. Membuatkan kasus khusus jika inputan n = 1 maka outputnya titik Convex Hull adalah titik itu sendiri
2. Untuk n>1,iterasi semua elemen dari array of Point yang berisi titik sembarang untuk mencari semua kemungkinan atau kombinasi pasangan yang dapat dibentuk di mana pasangan <(x1,y1);(x2,y2)> dianggap sama dengan <(x2,y2);(x1,y1)> sehingga total kemungkinan pasangan titik adalah **nC2 =** buah pasangan titik
3. Untuk setiap pasangan titik dalam n(n-1) kemungkinan tersebut dimisalkan pasangan titik tersebut membentuk suatu garis ax+by=c ; akan dihitungkan serta disimpan nilai a, b dan c ke dalam variable di mana nilai a = y2-y1 ; nilai b = x1-x2 ; dan nilai c = x1\*y2 – x2\*y1
4. Nilai a,b,dan c dari setiap pasangan titik tersebut akan digunakan sebagai acuan untuk mengetahui posisi nilai titik-titik lainnya terhadap garis yang dibentuk pasangan titik tersebut. Nilai a,b,c akan dimasukkan ke dalam persamaan ax+by-c dengan x dimasukkan nilai x3,x4,….xn ; y dimasukkan nilai y3,y4,……yn maka untuk setiap pasangan titik akan dicekkan dengan **(n-2)** titik lainnya. Garis ax+by=c akan membelah bidang koordinat 2 dimensi menjadi 2 bagian yaitu bidang di mana ax+by > 0 di satu sisi dan ax+by < 0 di sisi lain garis tersebut
5. Apabila nilai dari semua ax+by-c untuk semua titik selain pasangan titik tersebut bernilai positif atau 0(>=0) atau semua bernilai negatif atau nol(=<0) maka pasangan titik <(x1,y1);(x2,y2)> adalah titik pembentuk convex hull sebab garis yang dibentuk kedua titik memisahkan semua titik lainnya hanya di salah satu sisi (sisi ax+by-c>=0 atau sisi ax+by-c=<0). Apabila dalam proses iterasi ditemukan perbedaan tanda untuk nilai ax+by-c maka proses iterasi berhenti dan acuan pasangan titik akan berpindah ke pasangan titik lain seperti yang ada pada langkah 2
6. Untuk semua titik yang membentuk Convex Hull akan disimpan ke dalam 2 buah array. Array pertama merepresentasikan set of Convex Hull point yang elemen-elemennya **unik** yang akan ditampilkan sebagai output .Array kedua berisi semua titik yang memenuhi syarat pada langkah 5 sehingga array tersebut memiliki beberapa elemen yang merupakan titik yang sama. Array tersebut akan digunakan dalam proses penampilan grafik.

Algoritma Brute di atas memiliki Forces di atas memiliki **T(n) =() \*(n-2)** untuk operasi perbandingan nilai ax+by-c dan **T(n)=()\*(Neff1+Neff2)** untuk proses iterasi penyimpanan ke dalam array di mana Neff1,Neff2 masing-masing merepresentasikan jumlah elemen dalam set dan array ,serta array dan neff tersebut memenuhi **Neff1,Neff2 <= n** dan setsehingga kompleksitas untuk algoritma brute force di atas adalah **O(n3)**.

1. **Kode Program**

**2.1 Kode program adt.h**

File adt.h berisi adt point dan array of points serta primitif-primitif kedua adt tersebut yang digunakaan dalam mencari himpunan titik pembentuk Convex Hull

|  |
| --- |
| /\*ADT Points\*/  struct Point {    int X; /\* absis \*/    int Y; /\* ordinat \*/  };  /\*ADT ARRAY of Points\*/  struct Array {  struct Point TI[101];  int Neff;  };  #define Neff(T) (T).Neff  #define TI(T) (T).TI  #define Elmt(T,i) (T).TI[(i)]  #define ElmtX(T,i) (T).TI[(i)].X  #define ElmtY(T,i) (T).TI[(i)].Y  void MakeEmpty(struct Array \*T);  void TulisIsi (struct Array T);  int Search(struct Array T, struct Point P); // Mengecek apakah ada point P di dalam array T  void AddAsLastEl (struct Array \* T, struct Point P);  void AddElUnik (struct Array \* T, struct Point P);  void PrintPoint(struct Point P);  int ValueofA(struct Point acuan,struct Point random); //Mengkalkulasi nilai a dalam persamaan ax+by=c dari garis dibentuk 2 titik  int ValueofB(struct Point acuan,struct Point random);//Mengkalkulasi nilai b dalam persamaan ax+by=c dari garis dibentuk 2 titik  int ValueofC(struct Point acuan,struct Point random);//Mengkalkulasi nilai c dalam persamaan ax+by=c dari garis dibentuk 2 titik  int Calculate(int A,int B,int C,struct Point P);//Mengkalkulasi nilai Ax+By-C untuk titik P(x,y)  int IsAllSameSign (int counterPos, int counterMin);//Mengecek apakah counterPos == counterMin  void GenerateNPoint (struct Array \*T,int n);//Membentuk array of random point  void PrintGeneratedPoint (struct Array T,int n);  void ConvexHullPoints (struct Array \*CHull,struct Array \*ChullSet,struct Array arr\_P, int n);//menyimpan semua titik convex hull dari sejumlah titik sembarang |

**2.2 Kode program adt.cpp**

File adt.cpp berisi realisasi primitif-primitif adt yang telah dideklarasi di header file adt.h

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <math.h>  #include "adt.h"  void MakeEmpty(struct Array \*T){  Neff(\*T) = 0;  }  void TulisIsi (struct Array T){  printf("[");  for (int i=1;i<=Neff(T);i++){  printf(" (%d,%d) ",ElmtX(T,i),ElmtY(T,i));  }  printf("]");  }  int Search(struct Array T, struct Point P){  int i = 1;  int found=0;  while (i <= Neff(T) && (found==0)){  if ((ElmtX(T,i)==P.X) && (ElmtY(T,i)==P.Y)){  found = 1;  } else {  i ++;  }  }  return found;  }  void AddAsLastEl (struct Array \* T, struct Point P){  Neff(\*T) +=1;  ElmtX(\*T,Neff(\*T)) = P.X;  ElmtY(\*T,Neff(\*T)) = P.Y;  }  void AddElUnik (struct Array \* T, struct Point P){  if(! Search(\*T,P)){  AddAsLastEl (T,P);  }  }  void PrintPoint(struct Point P){  printf("(%d,%d)",P.X,P.Y);  }  int ValueofA(struct Point acuan,struct Point random){  return (random.Y-acuan.Y);  }  int ValueofB(struct Point acuan,struct Point random){  return (acuan.X-random.X);  }  int ValueofC(struct Point acuan,struct Point random){  return (acuan.X\*random.Y) - (acuan.Y\*random.X);  }  int Calculate(int A,int B,int C,struct Point P){//Calculate ax+by value  return (A\*P.X)+(B\*P.Y)-C;  }  int IsAllSameSign(int counterPos, int counterMin){  return (!((counterPos >0)&&(counterMin >0)));  }  void GenerateNPoint (struct Array \*T,int n){  for(int i=1;i<=n;i++){  ElmtX(\*T,i) = -100+(rand() % (200+1));  ElmtY(\*T,i) = -100+(rand() % (200+1));  Neff(\*T) += 1;  }  }  void PrintGeneratedPoint (struct Array T,int n){  for(int i=1;i<=n;i++){  PrintPoint(Elmt(T,i));  }  printf("\n");  }  void ConvexHullPoints(struct Array \*CHull,struct Array \*CHullSet, struct Array T, int n){  if (n==1){  AddElUnik(CHull,Elmt(T,1));  } else {  for (int i=1;i<=n-1;i++){  for (int j=i+1;j<=n;j++){  int A = ValueofA(Elmt(T,i),Elmt(T,j));  int B = ValueofB(Elmt(T,i),Elmt(T,j));  int C = ValueofC(Elmt(T,i),Elmt(T,j));  int k=1;  int counterPos=0;  int counterMin=0;  while ((k<=n)){  if ((k!=i) && (k!=j) && (IsAllSameSign(counterMin,counterPos))){  // if ((k!=i) && (k!=j)){  int val = Calculate(A,B,C,Elmt(T,k));  if (val > 0){  counterPos +=1;  }  if (val < 0){  counterMin +=1;  }  }  k++;  }  if ((counterMin== 0 ) || (counterPos==0)){//Memenuhi titik ke i dan j membentuk Convex Hull  AddAsLastEl(CHull,Elmt(T,i));  AddAsLastEl(CHull,Elmt(T,j));  AddElUnik(CHullSet,Elmt(T,i));  AddElUnik(CHullSet,Elmt(T,j));  }  }  }  }  } |

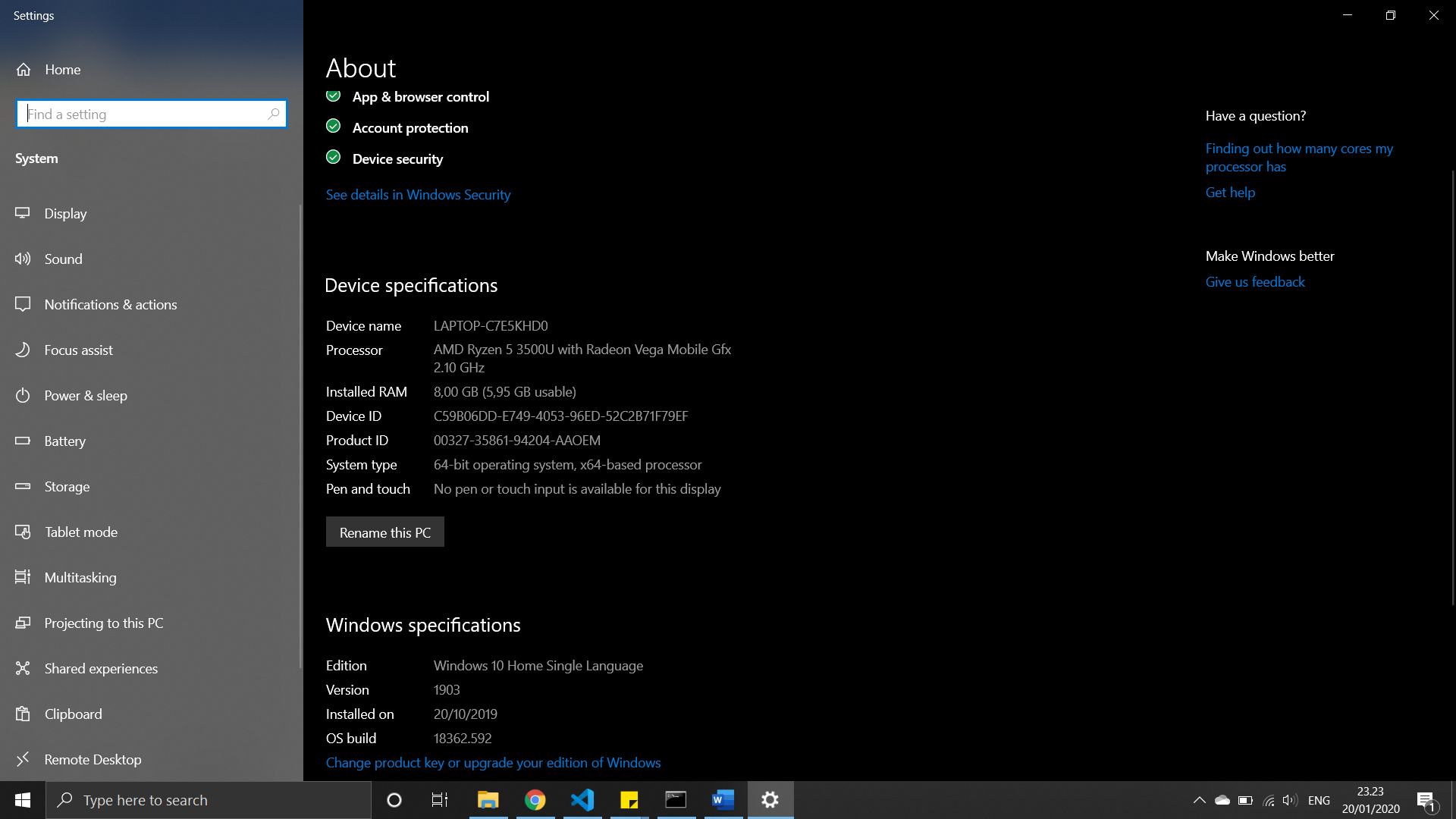
**2.3 Kode program main.c**

File main.c merupakan program utama yang menerima inputan jumlah titik dari pengguna, memanggil dan menjalankan semua primitif fungsi dari file adt.cpp untuk menampilkan output himpunan titik pembentuk Convex Hull serta menampilkan grafiknya.

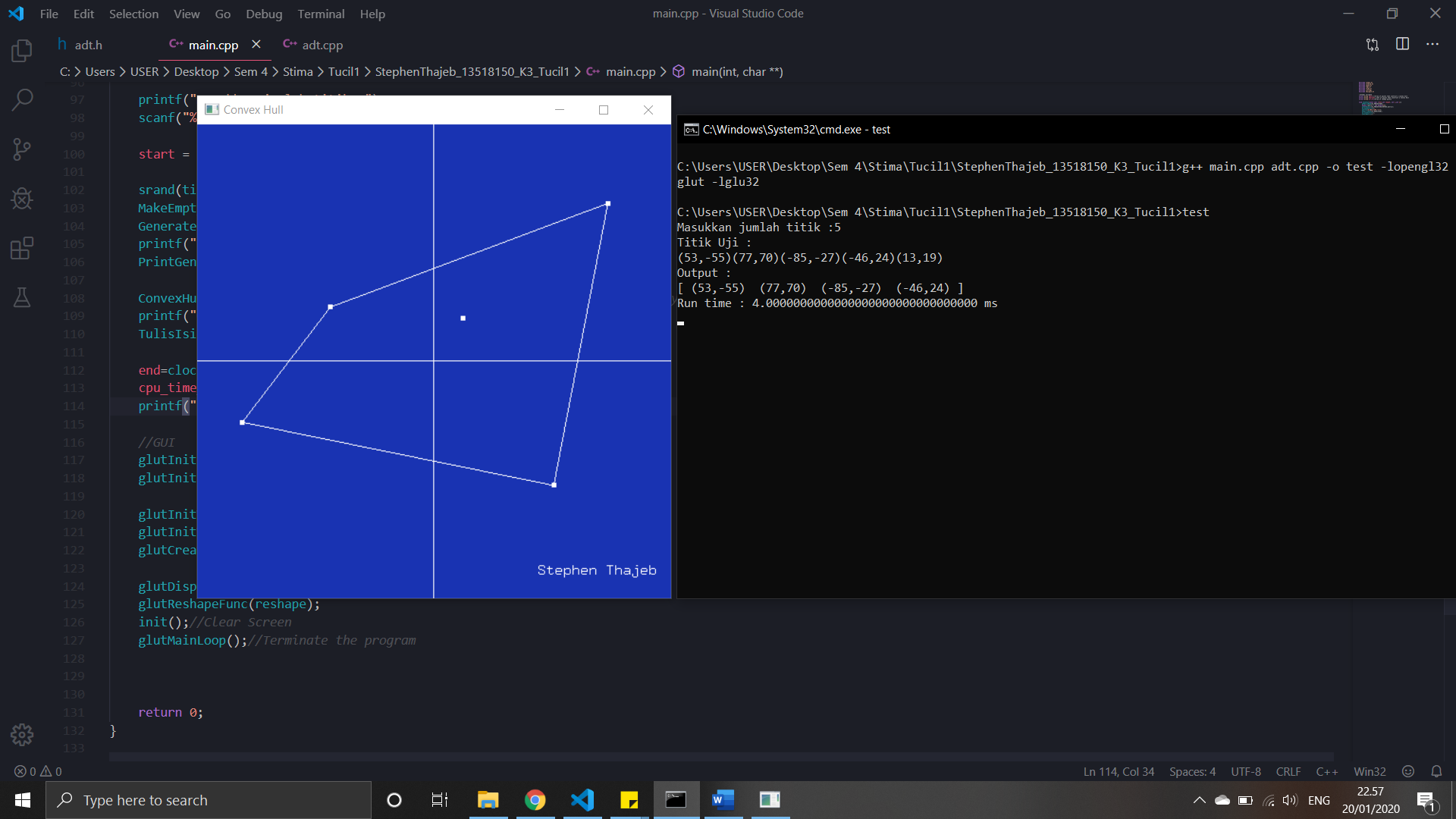
|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <math.h>  #include "adt.h"  #include "time.h"  #include <string>  #include <GL/glut.h>  //Global variable  struct Array CHull;//Array of points that contruct a convex hull  struct Array CHullSet;//Set of points that construct a Convex Hull  struct Array arr\_P;//Array of random points  void drawText(const char \*text,int length, int x,int y){  glMatrixMode(GL\_PROJECTION);  double \*matrix = new double[14];  glGetDoublev(GL\_PROJECTION\_MATRIX,matrix);  glLoadIdentity();  glOrtho(0,800,0,600,-5,5);  glMatrixMode(GL\_MODELVIEW);  glLoadIdentity();  glPushMatrix();  glLoadIdentity();  glRasterPos2i(x,y);  for(int i=0;i<length;i++){  glutBitmapCharacter(GLUT\_BITMAP\_9\_BY\_15,(int)text[i]);  }  glPopMatrix();  glMatrixMode(GL\_PROJECTION);  glLoadMatrixd(matrix);  glMatrixMode(GL\_MODELVIEW);  }  void display(void)  {  glClear(GL\_COLOR\_BUFFER\_BIT);//clear screen  glLoadIdentity();  glPointSize(5.0);  // draw  glBegin(GL\_LINES);  for (int i=1;i<=Neff(CHull)-1;i++){  if (i%2){  glVertex2i(ElmtX(CHull,i),ElmtY(CHull,i));  glVertex2i(ElmtX(CHull,i+1),ElmtY(CHull,i+1));  } else {  continue;  }    }  glEnd();  glBegin(GL\_LINES);  glVertex2i(0,90);  glVertex2i(0,-90);  glEnd();    glBegin(GL\_LINES);  glVertex2i(90,0);  glVertex2i(-90,0);  glEnd();  glBegin(GL\_POINTS);  for (int i=1;i<=Neff(arr\_P);i++){  glVertex2i(ElmtX(arr\_P,i),ElmtY(arr\_P,i));  }  glEnd();  std::string text1,text2;  text1 = "Stephen Thajeb";  text2 = "13518150";  drawText(text1.data(),text1.size(),575,30);  drawText(text2.data(),text2.size(),610,15);  glFlush();  }  void reshape(int w, int h)  {  glViewport(0, 0, (GLsizei)w, ( GLsizei)h);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-105,105,-105,105);  glMatrixMode(GL\_MODELVIEW);  }  void init(){  glClearColor(0.1, 0.2,0.7, 0.7);  }    int main (int argc,char\*\*argv){    int n;  clock\_t start,end;  double cpu\_time\_used;  printf("Masukkan jumlah titik :");  scanf("%d",&n);    start = clock();  srand(time(NULL));  MakeEmpty(&CHull);  GenerateNPoint(&arr\_P,n);  PrintGeneratedPoint(arr\_P,n);  ConvexHullPoints(&CHull,&CHullSet,arr\_P, n); //Mencari titik dalam arr\_P yang membentuk convex hull dan menyimpan di dalam array CHull dengan array berisi elemen unik  TulisIsi(CHullSet);//Menampilkan himpunan titik Convex Hull    end=clock();  cpu\_time\_used = ((double) (end - start)) / CLOCKS\_PER\_SEC;  printf("\nRun time : %.31f second\n", cpu\_time\_used);  //GUI  glutInit(&argc,argv);  glutInitDisplayMode(GLUT\_RGB);  glutInitWindowPosition(200,100);  glutInitWindowSize(500,500);  glutCreateWindow("Convex Hull");    glutDisplayFunc(display);  glutReshapeFunc(reshape);  init();//Clear Screen  glutMainLoop();//Terminate the program  return 0;  } |

1. **Screenshot Kode Program untuk n=5, 10, dan 20**

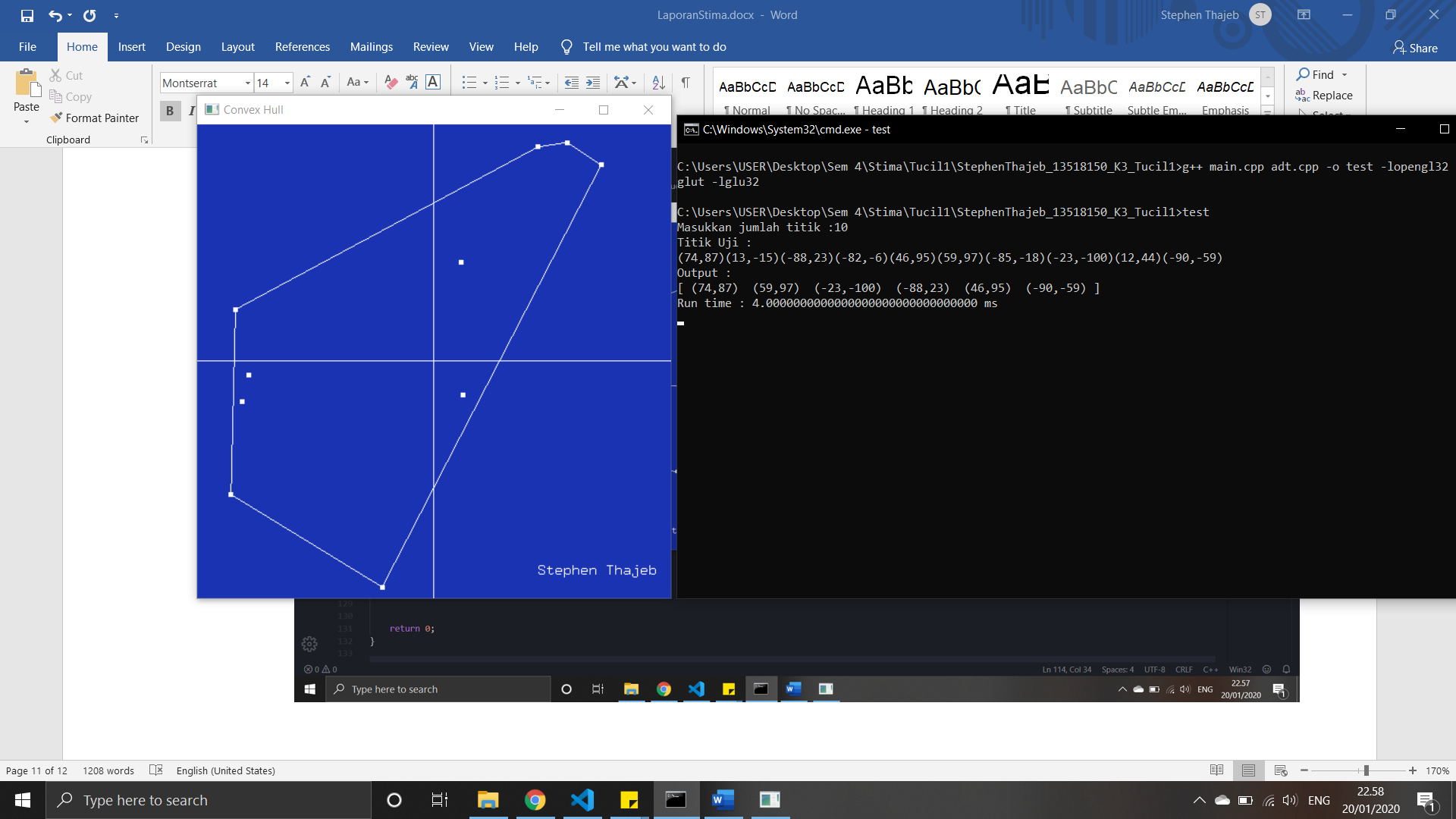
Hasil Program di bawah adalah hasil kompilasi dan running pada Laptop dengan spesifikasi di bawah ini



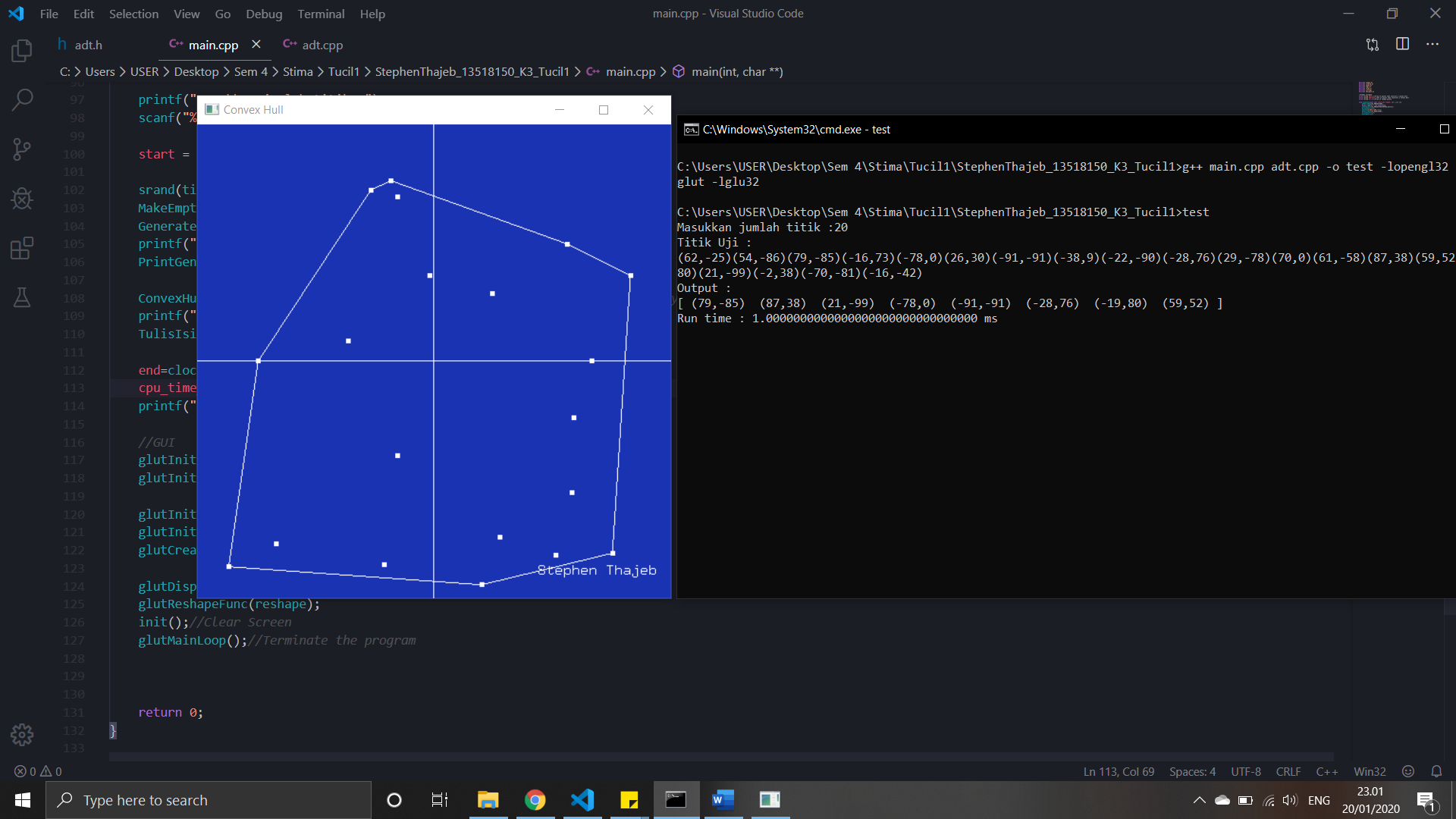
Gambar 1. Spesifikasi Laptop



Gambar 2. Tes Uji untuk n = 5



Gambar 3. Tes Uji untuk n = 10



Gambar 4. Tes Uji untuk n = 20

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Gambar 5. Hasil Uji untuk n = 5 | Gambar 6. Hasil Uji untuk n = 10 | Gambar 7. Hasil Uji untuk n = 20 |

|  |  |  |
| --- | --- | --- |
| Poin | Ya | Tidak |
| 1.Program berhasil dikompilasi | √ |  |
| 2.Program berhasil running | √ |  |
| 3.Program dapat menerima input dan output | √ |  |
| 4.Luaran sudah benar untuk semua n | √ |  |