

**Laporan Tugas Kecil 2**

**Implementasi Convex Hull untuk Visualisasi Tes Linear**

**Separability Dataset dengan Algoritma Divide and Conquer**

Mata Kuliah IF2211 - Strategi Algoritma



Oleh :

Nama : Roby Purnomo

NIM : 13520106

Kelas : 01

PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG

2021/2022

## Daftar Isi

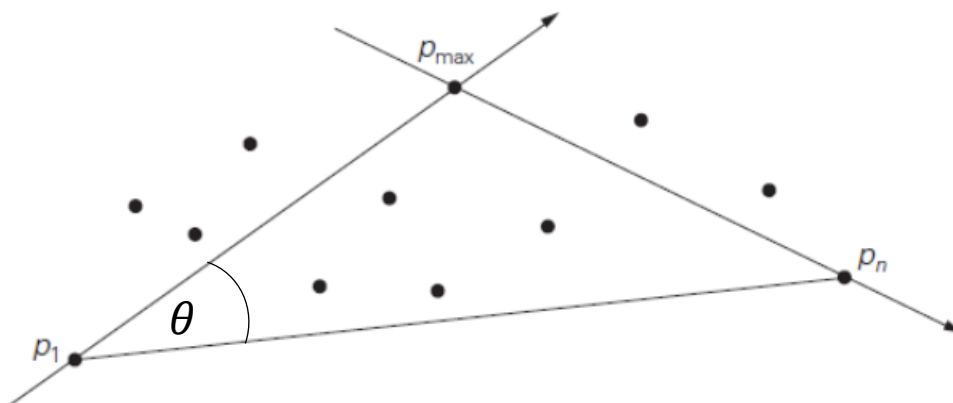
<b>Bab I : Teori Dasar dan Spesifikasi.....</b>	<b>2</b>
<b>Algoritma Divide and Conquer .....</b>	<b>2</b>
<b>Bab II : Source Program (Bahasa Python) .....</b>	<b>3</b>
<b>Modularity Program.....</b>	<b>3</b>
<b>ConvexHull.py .....</b>	<b>3</b>
<b>iris.py.....</b>	<b>5</b>
<b>main.py.....</b>	<b>6</b>
<b>Bab III : Testing.....</b>	<b>8</b>
<b>Main Program .....</b>	<b>8</b>
<b>Iris .....</b>	<b>8</b>
<b>breast_cancer.....</b>	<b>9</b>
<b>digits .....</b>	<b>9</b>
<b>linnerud.....</b>	<b>9</b>
<b>wine .....</b>	<b>10</b>
<b>Bab IV : Kesimpulan dan Saran.....</b>	<b>11</b>
<b>Kesimpulan.....</b>	<b>11</b>
<b>Saran .....</b>	<b>11</b>

# Bab I : Teori Dasar dan Spesifikasi

## Algoritma Divide and Conquer

Algoritma yang digunakan pada program ini yaitu mengimplementasikan strategi Divide and Conquer dengan langkah-langkah sebagai berikut.

1. Mula-mula cari titik yang terletak paling kiri dan paling kanan pada himpunan titik  $S$  dan simpan sebagai  $p_1$  dan  $p_n$
2. Lalu bagi menjadi 2 persoalan yaitu Convex Hull untuk titik di atas (top) garis  $p_1-p_n$  dan titik di bawah (bot) garis  $p_1-p_n$  yang dicek dengan menggunakan rumus determinan dan lalu membuat array baru dengan himpunan  $S$  yang memenuhi syarat untuk Convex Hull Top dan Bot tadi.
3. Pada Convex Hull Top dan Convex Hull Bot, cari titik ekstrem, yaitu titik terjauh dari garis  $p_1-p_n$  yang terletak di atas/kiri dari garis  $p_1-p_n$  untuk Convex Hull Top dan di bawah/kanan untuk Convex Hull Bot. Jika terdapat titik dengan jarak ke garis  $p_1-p_n$  sama, maka pilih titik yang memiliki sudut  $\theta$  pada gambar di bawah ini yang paling besar. Simpan titik ekstrem tersebut sebagai **pmax**.



4. Lalu dengan menggunakan prinsip divide and conquer, bagi permasalahan menjadi tingkat yang lebih kecil yaitu lakukan Convex Hull untuk bagian kiri dan kanan, yakni untuk garis  $p_1-p_{max}$  dan  $p_{max}-p_n$  dan menyimpan hasil return sebagai **left** dan **right**.
5. Lalu setelah melakukan divide, langkah terakhir adalah dengan melakukan merge yakni **left + pmax + right**, dan jika tidak ada nilai ekstremnya maka akan mengembalikan array kosong (sebagai basis)

## Bab II : Source Program (Bahasa Python)

Dapat dilihat secara full pada alamat github di bawah ini.

[https://github.com/robypurnomo/tucil\\_2\\_stima](https://github.com/robypurnomo/tucil_2_stima)

### Modularity Program

```
├── src
│   ├── lib
│   │   ├── convexHull
│   │   │   └── ConvexHull.py
│   │   ├── iris.py
│   │   ├── breast_cancer.py
│   │   ├── digits.py
│   │   ├── wine.py
│   │   └── linnerud.py
│   └── main.py
├── docs
│   ├── Laporan Tugas Kecil 2.pdf
│   └── Laporan Tugas Kecil 2.docx
├── testing
└── README.md
```

### ConvexHull.py

```
import numpy as np
from math import atan2, pi

# Mengecek letak titik pada zona atas atau bawah atau pada garis
def zoneCheck(p1, pn, p) :
    x1 = p1[0]
    x2 = pn[0]
    x3 = p[0]
    y1 = p1[1]
    y2 = pn[1]
    y3 = p[1]
    det = x1*y2 + x3*y1 + x2*y3 - x3*y2 - x2*y1 - x1*y3
    if det == 0 or x3 < x1 or x3 > x2 or (x3 == x1 and x3 == x2 and y3 ==
y1 and y3 == y2):
        return 0
    elif det > 0 :
```

```

        return 1
    else :
        return -1

# Menghitung jarak titik ke garis
def distance(pa, pb, px) :
    pa = np.array(pa)
    pb = np.array(pb)
    px = np.array(px)
    return abs(np.linalg.norm(np.cross(pb-pa, pa-px))/np.linalg.norm(pb-
pa))

# Menghitung sudut diantara tiga titik
def angle(p1, pmax, pn) :
    x1, y1 = p1[0] - pmax[0], p1[1] - pmax[1]
    x3, y3 = pn[0] - pmax[0], pn[1] - pmax[1]
    a = atan2(y1, x1)
    c = atan2(y3, x3)
    if a < 0: a += pi*2
    if c < 0: c += pi*2
    return (pi*2 + c - a) if a > c else (c - a)

# ConvexHull Recursive
def ConvexHullRec(bucket, p1, pn, value) :
    idx = -1
    dis = 0
    for i in range (len(bucket)) :
        if distance(pn, p1, bucket[i]) > dis or (distance(pn, p1,
bucket[i]) == dis and dis != 0 and angle(p1, bucket[i], pn) > angle(p1,
bucket[idx], pn)) :
            dis = distance(pn, p1, bucket[i])
            idx = i
    if idx != -1 :
        bucketleft = []
        bucketright = []
        for i in range (len(bucket)) :
            if zoneCheck(p1, bucket[idx], bucket[i]) == value :
                bucketleft.append(bucket[i])
            if zoneCheck(bucket[idx], pn, bucket[i]) == value :
                bucketright.append(bucket[i])
        left = ConvexHullRec(bucketleft, p1, bucket[idx], value)
        right = ConvexHullRec(bucketright, bucket[idx], pn, value)
        return left + [bucket[idx]] + right
    else :
        return []

# Implementasi ConvexHull
def myConvexHull(bucket) :

```

```

bucketlist = bucket.tolist()

p1 = bucketlist[0]
pn = bucketlist[0]
for i in range(len(bucketlist)) :
    if (bucketlist[i][0] < p1[0]) :
        p1 = bucketlist[i]
    elif (bucketlist[i][0] == p1[0] and bucketlist[i][1] < p1[0]) :
        p1 = bucketlist[i]
    if (bucketlist[i][0] > pn[0]) :
        pn = bucketlist[i]
    elif (bucketlist[i][0] == pn[0] and bucketlist[i][1] > pn[0]) :
        pn = bucketlist[i]

buckettop = []
bucketbot = []
for i in range (len(bucket)) :
    if zoneCheck(p1, pn, bucket[i]) == 1 :
        buckettop.append(bucket[i])
    elif zoneCheck(p1, pn, bucket[i]) == -1 :
        bucketbot.append(bucket[i])

array_point_top = [p1]
array_point_bot = [pn]

top = ConvexHullRec(buckettop, p1, pn, 1)
bot = ConvexHullRec(bucketbot, p1, pn, -1)

array_point_top += top + [pn]
array_point_bot += bot + [pn]

x = []
y = []

for i in range (len(array_point_top)) :
    x.append(array_point_top[i][0])
    y.append(array_point_top[i][1])
for i in range (len(array_point_bot)-1, -1, -1) :
    x.append(array_point_bot[i][0])
    y.append(array_point_bot[i][1])

return x,y

```

**iris.py**

```
import pandas as pd
```

```

import matplotlib.pyplot as plt
from sklearn import datasets
from lib.convexHull.ConvexHull import myConvexHull

def iris() :

    data = datasets.load_iris()

    # create a DataFrame
    df = pd.DataFrame(data.data, columns=data.feature_names)
    df['Target'] = pd.DataFrame(data.target)
    print(df.shape)
    df.head()

    # visualisasi ConvexHull
    plt.figure(figsize = (10, 6))
    colors = ['b', 'r', 'g']
    plt.title('mean radius vs mean texture')
    plt.xlabel(data.feature_names[0])
    plt.ylabel(data.feature_names[1])
    for i in range(len(data.target_names)):
        bucket = df[df['Target'] == i]
        bucket = bucket.iloc[:, [0, 1]].values
        x, y = myConvexHull(bucket)
        plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
        plt.plot(x, y, colors[i])
    plt.legend()
    plt.show()

```

## main.py

```

from lib.iris import iris
from lib.digits import digits
from lib.linnerud import linnerud
from lib.wine import wine
from lib.breast_cancer import breast_cancer

while (True) :

    print("Data mana yang ingin anda visualisasikan convex hull-nya ?")
    print("1. Iris")
    print("2. Breast_Cancer")
    print("3. Digits")
    print("4. Linnerud")
    print("5. Wine")
    print("6. Exit")

```

```
print()

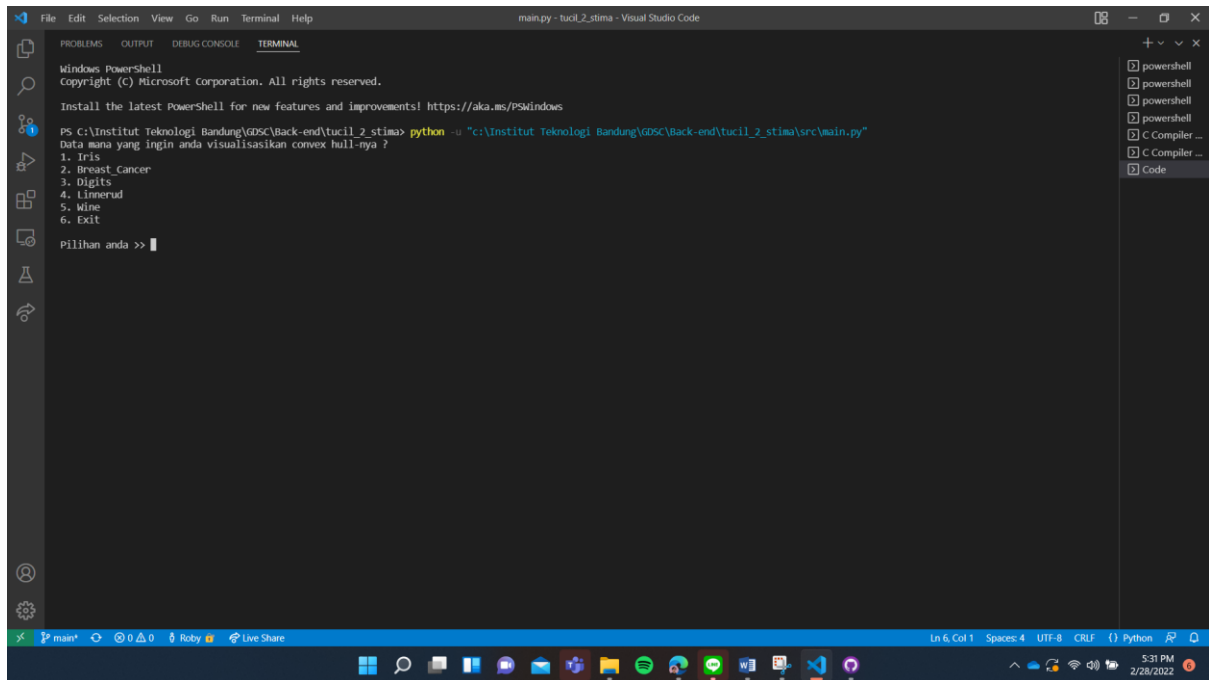
choice = int(input("Pilihan anda >> "))

if choice == 1:
    iris()
elif choice == 2:
    breast_cancer()
elif choice == 3:
    digits()
elif choice == 4:
    linnerud()
elif choice == 5:
    wine()
elif choice == 6:
    break
```



# Bab III : Testing

## Main Program



```
File Edit Selection View Go Run Terminal Help
main.py - tucil_2_stima - Visual Studio Code

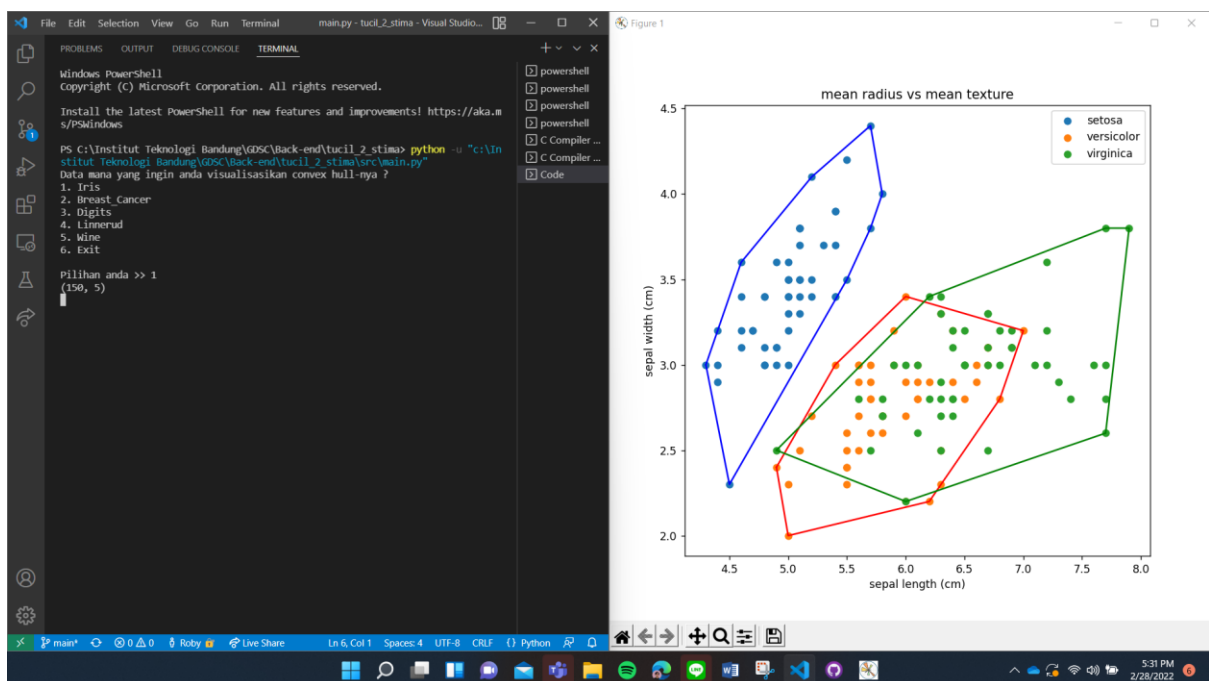
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

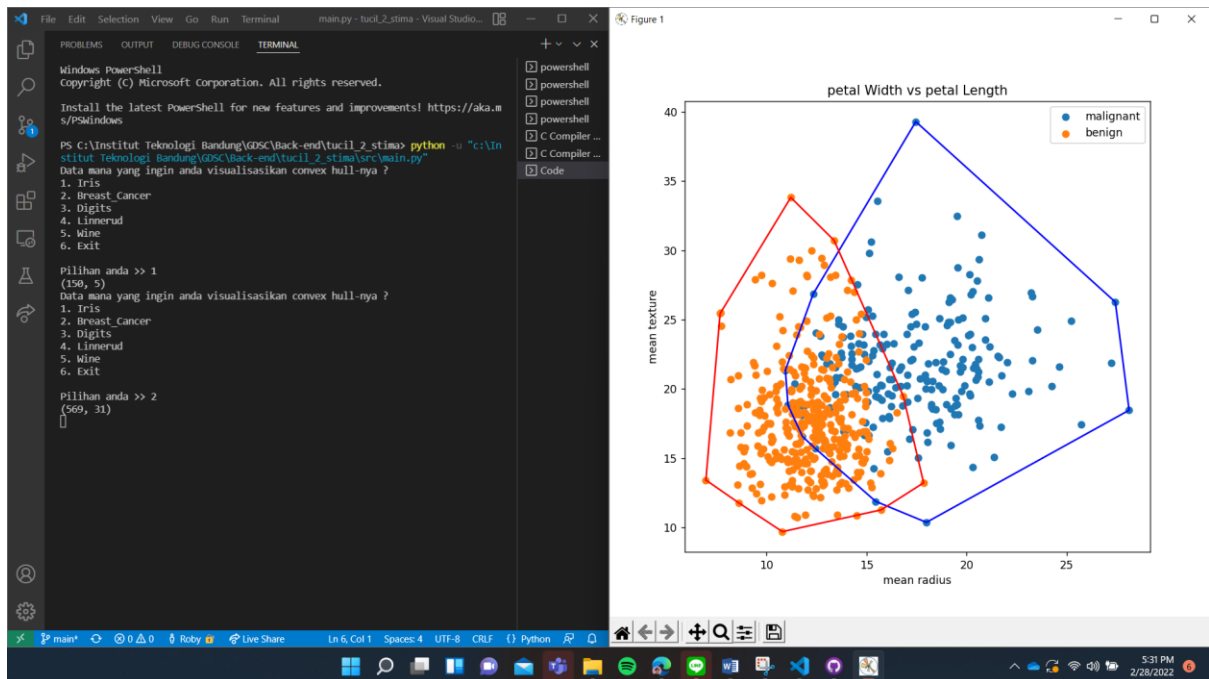
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Institit Teknologi Bandung\GOSC\Back-end\tucil_2_stima> python -u "c:\Institit Teknologi Bandung\GOSC\Back-end\tucil_2_stima\src\main.py"
Data mana yang ingin anda visualisasikan convex hull-nya ?
1. Iris
2. Breast_Cancer
3. Digits
4. Linnerud
5. Wine
6. Exit
Pilihan anda >> 1
```

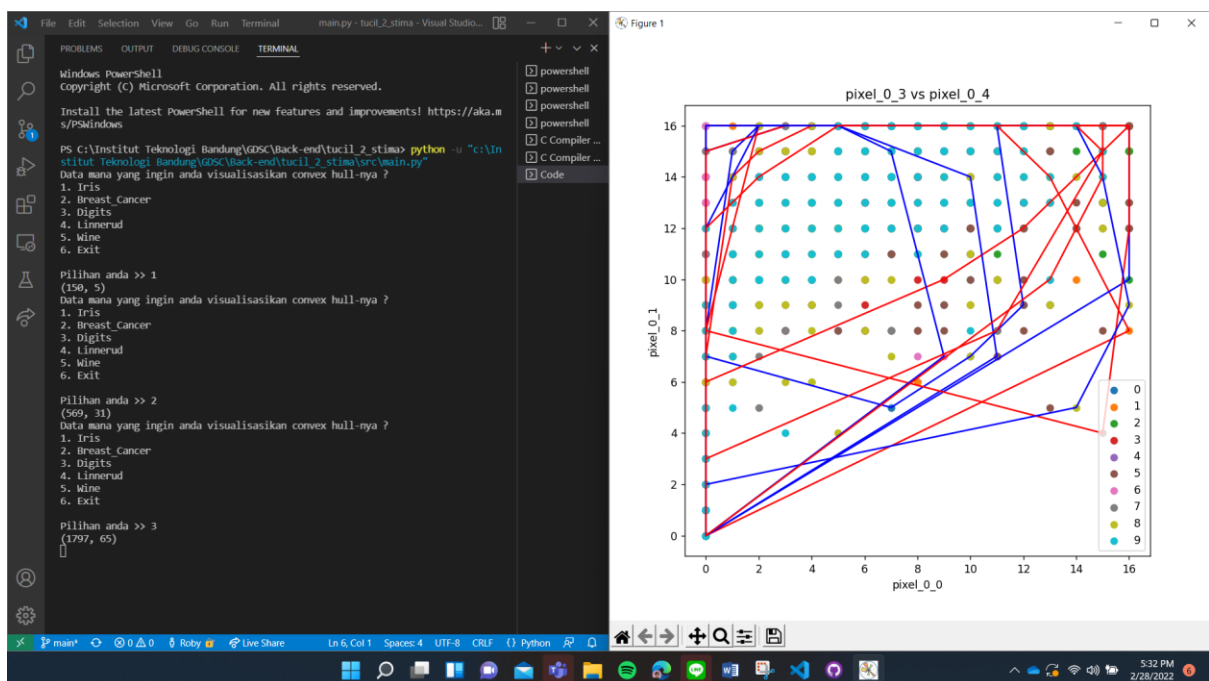
## Iris



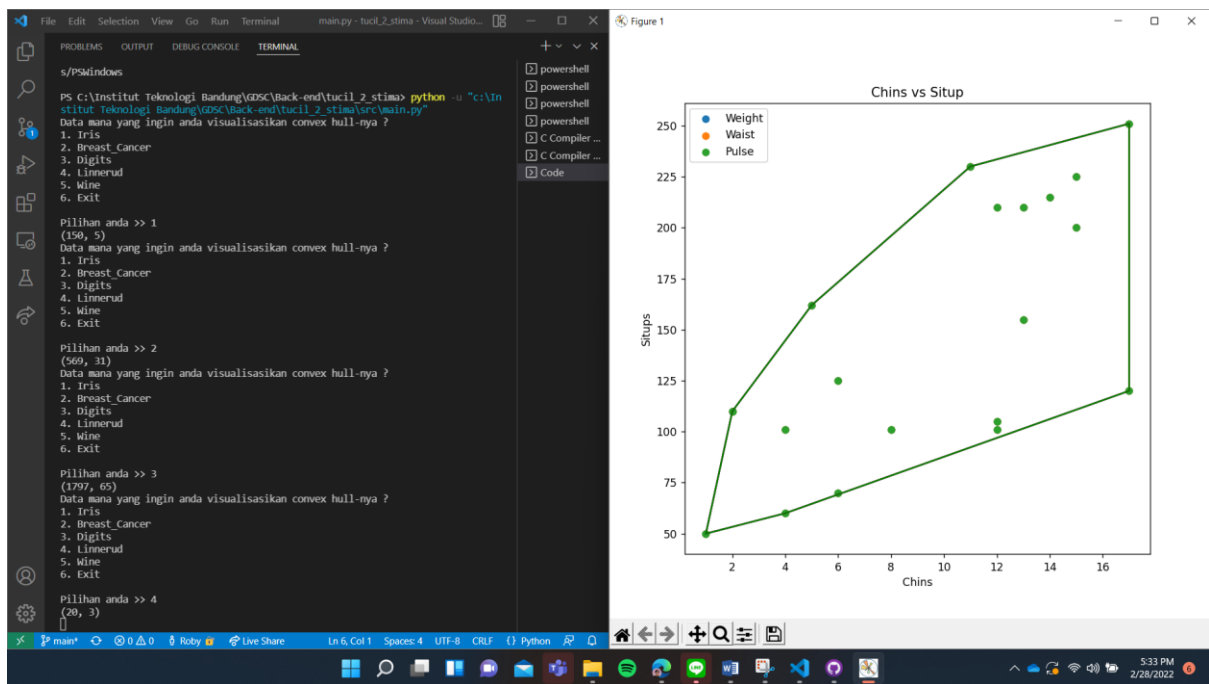
## breast\_cancer



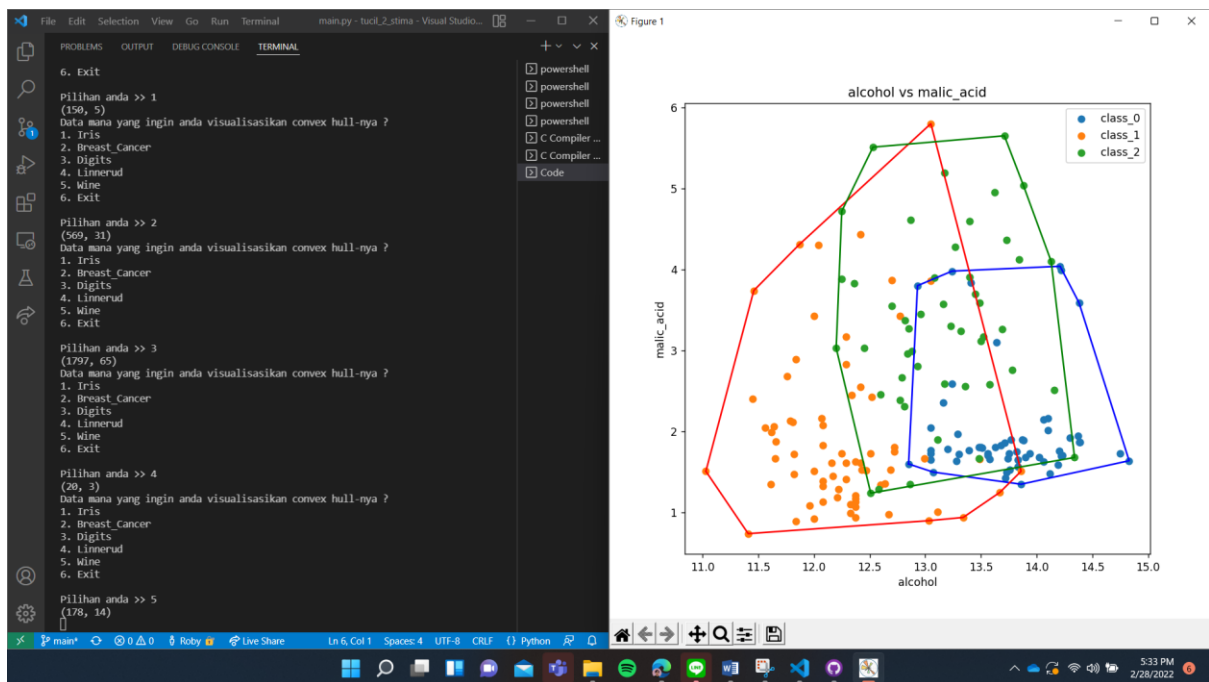
## digits



## linnerud



wine



## Bab IV : Kesimpulan dan Saran

### Kesimpulan

Saya berhasil membuat Implementasi Convex Hull untuk “**Visualisasi Tes Linear Separability Dataset dengan Algoritma Divide and Conquer**” dengan bahasa pemrograman python dengan spesifikasi dan sejauh ini belum ada solusi dari program yang terlewat dan/atau tidak tepat. Dan juga saya melakukan testing untuk sampel data lain yaitu iris, wine, breast\_cancer, digits, dan linnerud

Poin	Ya	Tidak
1. Pustaka myConvexHull berhasil dibuat dan tidak ada kesalahan	✓	✗
2. Convex hull yang dihasilkan sudah benar	✓	✗
3. Pustaka myConvexHull dapat digunakan untuk menampilkan convex hull setiap label dengan warna yang berbeda.	✓	✗
4. <b>Bonus</b> : program dapat menerima input dan menuliskan output untuk dataset lainnya	✓	✗

### Saran

Kedepannya mungkin dapat ditemukan algoritma yang lebih baik daripada algoritma Divide and Conquer yang diimplementasikan pada program kali ini dalam segi kompleksitas maupun efektifitas.