LAPORAN TUGAS KECIL

Implementasi Convex Hull untuk Visualisasi Tes Linear Separability Dataset dengan Algoritma Divide and Conquer

Disajikan untuk memenuhi salah satu tugas kecil Mata Kuliah IF2211 Strategi Algoritma yang diampu oleh:

Dr. Masayu Leylia Khodra, S.T., M.T.

Disusun oleh:

Nelsen Putra (13520130)

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

Daftar Isi

BAB I
Algoritma Divide and Conquer

Penjelasan Algoritma Divide and Conquer yang digunakan:

- Program dimulai dengan pembacaan dataset yang akan menghasilkan kumpulan titik-titik pada setiap *convex hull* yang bersesuaian yang masing-masing telah dibedakan warnanya dan disimpan dalam suatu array bucket.
- Hal pertama yang dilakukan adalah mencari dua titik terluar dari titik-titik yang terdapat pada dataset, dua titik ini dinamakan sebagai p1 dan pn.
- Titik p1 ditentukan dengan mencari nilai x minimal pada bucket sedangkan titik pn ditentukan dengan mencari nilai x maksimal pada bucket yang dapat dicari dan ditentukan dengan pemanggilan fungsi extremeIndex(bucket) yang akan mengembalikan nilai x pada nilai minimalnya dan nilai x pada nilai maksimalnya. Selanjutnya, akan dibentuk titik p1 dari masukan nilai x minimal dan y yang bersesuaian dengannya dan terbentuk titik pn dari masukan x maksimal dan y yang bersesuaian dengannya.
- Setelah didapatkan dua buah titik tersebut, maka kedua titik, yakni p1 dan pn dapat dihubungkan menjadi garis yang akan memisahkan titik-titik bucket menjadi bagian kiri dan kanan.
- Selanjutnya, dilakukan pemeriksaan apakah suatu titik berada pada sebelah kiri (atas) atau sebelah kanan (bawah) garis p1pn.
- Pengecekan dilakukan dengan melakukan pemanggilan fungsi pointPosition(p1, pn, check, bucket) yang akan memanggil fungsi determinan(p1, pn, check, bucket) yang akan mengembalikan determinan dari kumpulan titik p1, pn, dan cek pada bucket. Apabila determinan > 0, maka titik-titik tersebut akan berada pada bagian kiri dari garis p1pn dan berada pada bagian kanan apabila sebaliknya.
- Selanjutnya, setelah pemeriksaan, proses akan memasuki bagian *divide and conquer* pada fungsi rekursif, yakni recursiveFunction(p1, pn, bucketTrue, bucket) dimana program akan mulai melakukan pengecekan terhadap jumlah titik yang berada pada bucket, yaitu titik selain p1pn.
- Jika tidak ada titik pada bucket, maka akan dilakukan pengecekan terhadap titik p1pn apakah terbentuk garis atau tidak, jika p1 != pn, akan mengembalikan array berupa titik [p1,pn], sedangkan apabila p1 = pn, akan dikembalikan array kosong.
- Apabila jumlah titik-titik pada bucket > 0, akan dilakukan pengecekan lebih lanjut, yaitu dengan cara memilih titik yang memiliki jarak terjauh dari garis p1pn, misalnya px. Jika terdapat titik dengan jarak yang sama, maka yang diambil adalah titik yang memaksimalkan p1pxpn dimana pengecekan sudutnya dapat dilakukan dengan pemanggilan fungsi trianglePartition(p1, p2, p3) yang parameternya merepresentasikan ketiga titik tersebut.

- Selanjutnya, dilakukan pembagian pengecekan yaitu untuk bagian sebelah kiri dan sebelah kanan. Bagian sebelah kiri dilakukan dengan pengecekan kumpulan titik yang berada pada sebelah kiri p1px. Sedangkan bagian sebelah kanan, dilakukan pengecekan pada kumpulan titik yang berada pada sebelah kanan pxpn.
- Hal ini dilakukan secara rekursif hingga seluruh titik pada sebelah kiri dan kanannya telah selesai diperiksa.
- Setelah semua titik pada bagian kiri dan kanan selesai diperiksa, maka tahap terakhir yang dilakukan adalah melakukan *combine* antara fungsi rekursif pada bagian kiri dan fungsi rekursif pada bagian kanan. Setelah itu, akan dilakukan pengecekan kembali hingga program berakhir dan didapatkan *convex hull* final seperti yang diharapkan.

BAB II
Source Code Program

Program dibuat dengan menggunakan bahasa Python dengan memanfaatkan *library* matplotlib pada Python untuk memproyeksikan *convex hull* yang didapatkan nantinya. Struktur program myConvexHull adalah sebagai berikut.

1. Folder output
   Berisikan gambar dari hasil testing program myConvexHull.
2. Folder test
   Berisikan file test.txt yang memberikan penjelasan mengenai dataset apa saja yang digunakan dan diuji pada program ini.
3. Folder src
   Berisikan *source code* program utama yang terdiri atas:
   1. File function.py
      Berisikan fungsi-fungsi dasar pembentuk *convex hull* sebagai berikut:
      - def determinan(p1, pn, check, bucket)
      - def pointPosition(p1, pn, check, bucket)
      - def extremeIndex(bucket)
      - def trianglePartition(p1, p2, p3)
      - def recursiveFunction(p1, pn, bucketTrue, bucket)

   2. File myConvexHull.py
      Berisikan fungsi utama yaitu def myConvexHull(bucket)

   3. File main.py
      Digunakan untuk *testing* program dengan bahasa Python dan berisikan tampilan utama program.

   4. File visualization.ipynb
      Berisikan visualisasi hasil *testing* program dari 4 dataset, yaitu dataset iris, dataset breast_cancer, dataset digits, dan dataset wine.
4. Folder doc
   Berisikan laporan tugas kecil dengan isi dan format sesuai dengan ketentuan yang telah diberikan.
5. README.md
   Berisikan tata cara penggunaan program yang minimal berisi deskripsi singkat program, *requirement* dan instalasi modul program, langkah meng-*compile* program, cara menggunakan, serta identitas pembuat program.

Secara lengkap, berikut *source code* dari program myConvexHull dengan bahasa Python.

1. file Function.py

```
# function.py
# Source code program yang berisi fungsi-fungsi dasar pembentuk fungsi myConvexHull

# IMPORT LIBRARY
import numpy as np


'''
/* *** DETERMINAN *** */
** Mengembalikan determinan dari kumpulan tiga titik yang berada dalam array bucket
** param: p1, pn, check (titik-titik yang akan dicari determinannya)
** param: bucket (array tempat titik berada)
** return: det (determinan dari ketiga titik)
'''
def determinan(p1, pn, check, bucket):
    # Inisialisasi semua elemen dari titik-titiknya
    x1 = bucket[p1][0]
    y1 = bucket[p1][1]
    x2 = bucket[pn][0]
    y2 = bucket[pn][1]
    x3 = bucket[check][0]
    y3 = bucket[check][1]

    # Menghitung determinan
    det = (x1 * y2) + (x3 * y1) + (x2 * y3) - (x3 * y2) - (x2 * y1) - (x1 * y3)
    return det


'''
/* *** POINT POSITION *** */
** Mengecek posisi titik, yakni berada di kiri/atas atau kanan/bawah dari garis yang merupakan perpanjangan
dari titik p1 dan pn
** param: p1, pn (titik-titik terluar dari kumpulan titik yang kemudian membentuk garis)
** param: check (titik yang diperiksa)
** param: bucket (array tempat titik berada)
** return: determinan dari ketiga titik yang kemudian digunakan untuk menentukan posisi titik
'''
def pointPosition(p1, pn, check, bucket):
    # Jika determinan > 0 (positif), titik berada di sebelah kiri/atas garis
    # Jika determinan <= 0 (negatif), titik berada di sebelah kanan/bawah garis
    return determinan(p1, pn, check, bucket)


'''
/* *** EXTREME INDEX *** */
```

```
** Mengembalikan indeks dari titik koordinat pada bucket yang memiliki nilai x minimum atau maksimum
** param: bucket (array tempat titik berada)
** return: indeks minimum dan maksimum dari titik pada bucket
'''
def extremeIndex(bucket):
    # Inisialisasi nilai x
    valueOfX = []

    # Menambahkan semua elemen bucket ke dalam nilai x
    for i in range(len(bucket)):
        valueOfX.append(bucket[i][0])
    maxOfX = max(valueOfX)
    minOfX = min(valueOfX)

    # Pengecekan untuk indeks maksimum
    found = True
    indexMaxOfBucket = 0
    while ((indexMaxOfBucket < len(bucket)) and (found)):
        if (bucket[indexMaxOfBucket][0] == maxOfX):
            found = False
        else:
            indexMaxOfBucket += 1

    # Pengecekan untuk indeks minimum
    found = True
    indexMinOfBucket = 0
    while ((indexMinOfBucket < len(bucket)) and (found)):
        if (bucket[indexMinOfBucket][0] == minOfX):
            found = False
        else:
             indexMinOfBucket += 1

    # Mengembalikan indeks minimum dan indeks maksimum dari titik pada bucket
    return (indexMinOfBucket, indexMaxOfBucket)


'''
/* *** TRIANGLE PARTITION *** */
** Mengembalikan besar sudur apit dari partisi segitiga
** param: p1, p3 (titik-titik terluar dari kumpulan titik yang kemudian membentuk garis))
** param: p2 (titik periksa)
** return: degree (sudut apit dari partisi segitiga)
'''
def trianglePartition(p1, p2, p3):
    # Menghitung jarak antara titik p1-p2 dan p3-p2
    distance12 = p1 - p2
    distance32 = p3 - p2
    # Menghitung dot product antara vektor distance12 dan distance32
```

```
    product1 = np.dot(distance12, distance32)
    # Menghitung vektor normal dari distance12 dan distance32
    product2 = np.linalg.norm(distance12) * np.linalg.norm(distance32)

    # Menghitung besar sudut p2 (sudut apit dari partisi segitiga)
    cosinusValue = product1 / product2
    degree = np.degrees(np.arccos(cosinusValue))
    return degree


'''
/* *** RECURSIVE FUNCTION *** */
** Fungsi yang akan digunakan secara rekursif pada fungsi myConvexHull
** param: p1, pn (titik-titik terluar dari kumpulan titik yang kemudian membentuk garis)
** param: bucketTrue (array yang berisi titik-titik pada convex hull)
** param: bucket (array yang berisi semua titik)
** return: array pada bucket yang bernilai true
'''
def recursiveFunction(p1, pn, bucketTrue, bucket):
    # Kondisi banyak titik pada bucket != 0
    if (len(bucket) != 0):
        # Inisialisasi array kosong
        temp = []

        # Pengecekan semua titik pada bucket
        for i in range(len(bucket)):
            if ((p1 != pn) and (p1 != bucket[i]) and (pn != bucket[i])):
                degreeOfTemp = trianglePartition(bucketTrue[pn], bucketTrue[p1], bucketTrue[bucket[i]])
            else:
                degreeOfTemp = 0
            # Menambahkan sudut temp ke array temp
            temp.append(degreeOfTemp)
        # Inisialisasi titik px dengan bucket pada index ke max dari array temp
        px = bucket[temp.index(max(temp))]

        # DIVIDE AND CONQUER SECARA REKURSIF
        # Inisialisasi array yang berisi titik pada garis Px-Pn
        pointOfPxPn = []
        # Pengecekan
        for i in range(len(bucket)):
            if ((pointPosition(px, pn, bucket[i],bucketTrue) > 0) and (bucket[i] != p1) and (bucket[i] !=
pn)):
                pointOfPxPn.append(bucket[i])

        # Inisialisasi array yang berisi titik pada garis P1-Px
        pointOfP1Px = []
        # Pengecekan
        for i in range(len(bucket)):
```

```
            if ((pointPosition(p1, px, bucket[i], bucketTrue) > 0) and (bucket[i] != p1) and (bucket[i] !=
pn)):
                pointOfP1Px.append(bucket[i])

        # Pemanggilan fungsi rekursif kembali
        left = recursiveFunction(p1, px, bucketTrue, pointOfP1Px)
        right = recursiveFunction(px, pn, bucketTrue, pointOfPxPn)

        # Mengembalikan hasil kombinasi dari partisi yang sudah dilakukan sebelumnya berupa array dari bucket
yang bernilai true
        return left + right

    # Kondisi banyak titik pada bucket = 0
    else:
        # Pengecekan apakah p1 dan pn merupakan titik yang berbeda agar dapat membentuk garis
        if (p1 != pn):
            return [[p1, pn]]
        else:
            return []
```

2. file myConvexHull.py

```
# myConvexHull.py
# Source code program yang berisi implementasi fungsi utama myConvexHull

# IMPORT LIBRARY AND FUNCTION
import numpy as np
from function import *


'''
/* *** myConvexHull *** */
** Mengembalikan semua titik-titik yang merupakan titik terluar dari convex hull
** param: bucket (array of titik convex hull yang diuji)
** return: kumpulan titik terluar dar convex hull
'''
def myConvexHull(bucket):
    # Membagi kumpulan titik menjadi 2 bagian, yakni titik bagian kiri dan titik bagian kanan
    leftBucket = []
    rightBucket = []

    # Mendaftarkan titik-titik pada array bucket
    bucketTrue = np.array(bucket).astype(float)

    # Mencari titik p1 dan pn yang merupakan titik terluar pada bucket
    p1, pn = extremeIndex(bucketTrue)
```

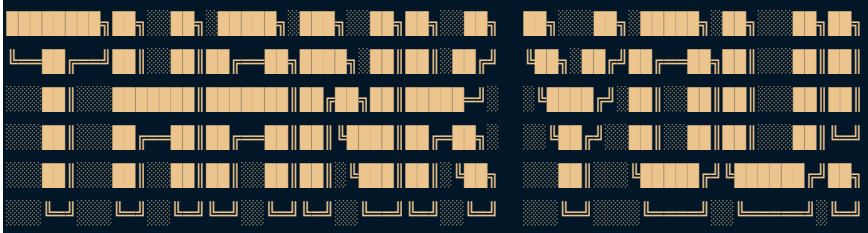
```
    # Pengecekan posisi seluruh titik
    for i in range(len(bucketTrue)):
        if ((pointPosition(p1, pn, i, bucketTrue) > 0) and (i != p1) and (i != pn)):
            leftBucket.append(i)
        elif ((pointPosition(p1, pn, i, bucketTrue) < 0) and (i != p1) and (i != pn)):
            rightBucket.append(i)

    # Memanggil fungsi rekursif untuk mendapatkan titik-titik yang sesuai
    left = recursiveFunction(p1, pn, bucketTrue, leftBucket)
    right = recursiveFunction(pn, p1, bucketTrue, rightBucket)

    # Mengembalikan hasil kombinasi dari partisi yang sudah dilakukan sebelumnya berupa array dari bucket
yang bernilai true
    return left + right
```

3. File main.py

```python
# main.py
# Source code program utama

# IMPORT LIBRARY DAN MYCONVEXHULL
import pandas as pd
import matplotlib.pyplot as plt
import random
from scipy.spatial import ConvexHull
from myConvexHull import myConvexHull

def header():
    print("[]============================================================================================[]")
    print("||    _ _            _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _      _ _ _ _ _ _ _ _ _  ||")
    print("||   /   |          |   \    __ _|   |    _ _ _ _|    \    \    /    |    __ _ _|  ||")
    print("||  /   /    _ __    \   \  /_ __    |   | /   _ _ _   \    \_ _/     |   /_ _ __   ||")
    print("||  \   \    /   \    /   /       |   |   | /   /    \   \            |          |  ||")
    print("||   \   \  /  /\  \  /   /   _ _ __|   |   | \   \_ _ _/   / |\      /|   |    _ _ _ _|  ||")
    print("||    \   \/  /__\  \/   /    \_ _ _    |_ _|_ \_ __      /  | \_ _ _/ |   |    \_ _ __   ||")
    print("||     \_ _ _ _ _ _ _ __/_ _ _ _ _ _|_ _ _ _ _|_ _ _|_ _ _/_ _|        |_ _|__ _ _ _ _ |  ||")
    print("||                                _ _ _ _ _ ___ _ _ _ _                                    ||")
    print("||                              |_ __    _ __|       \                                    ||")
    print("||                                  |  |  /   _ _    \                                   ||")
    print("||                                  |  | /    /   \    \                                 ||")
    print("||                                  |  | \    \_ _/    /                                 ||")
    print("||                                  |  |  \           /                                  ||")
    print("||                                  |__|   \ _ _ _ __/                                   ||")
    print("||   _ _ _ _ _ _ _ _ _ _      _ _ _           _ _ _ _ _ _ _ _       _ _ _               ||")
    print("||  |    _ _ _|      \    \    |   |  \        /    /  _ _ _ |\     /    /               ||")
    print("||  |    |   /   _ _   \    \   |   |   \      /    /  /_ _\__  \_ _/    /                ||")
    print("||  |    | /   /    \   \    \  |   |    \    /    /          | _ _    /                  ||")
    print("||  |    | \   \_ _/   /      \ |   |\    \_ /    /     _ _ __| /   \   \                 ||")
    print("||  |    |_ _\_        /        \|   | \          /     \_ _/__ /     \   \               ||")
    print("||  |_ _ _ _ _|_ _ _ /|_ _|\_ _ _ _|  \_ _ _ __/_ _ _ _ _ _ _ _|_ _     \_ _\_ _ _ _ _    ||")
    print("||                                          _ _ __       _ _ _ _       _ _ _ _ _ _ _       ||")
    print("||                                        |    |    |   |   |     |   |   |   |       ||")
    print("||                                        |    |    |   |   |     |   |   |   |       ||")
    print("||                                        |    |_ __|   |   |     |   |   |   |       ||")
    print("||                                        |     _ __    |   |     |   |   |   |       ||")
    print("||                                        |    |    |   |   |_ _ _|   |   |   |       ||")
    print("||                                        |    |    |   |             |   |_ _|_ _ _   ||")
    print("||                                        |_ __|    |_ _|_ _ _ _ _ _ _|_ _ _ _ _|_ _| ||")
    print("||                  _ _ __ __ _ _ _ _ _ __ _ _        _ _ _ _ _ __ _ _ _ _ _             ||")
    print("||              /    _ _ _|       \    |   \       /    /  _ _ _|   _ _   |              ||")
    print("||             /    /   /    _ _   \   |    \     /    /  /_ _ _    |_ _|  |              ||")
    print("||             \    \  /    /   \   \  |\    \   /    /         |        /               ||")
    print("||              \    \ \    \_ _/   /  | \    \_/    /    _ _ __|    |\   \              ||")
    print("||           __ _\    \ \          /   |_ \_ __     /     \_ _ _     | \   \             ||")
```

```
    print("||          |_ _ _ _ _/  \_ _ _ _ /_ _ _ _ _ _ _|_ _ /_ _ _ _ _ _ _ |_ _|  \_ _\             ||")
    print("||                                                                                             ||")
    print("[]=============================================================================================[]")
    print("[]============================Created by Nelsen Putra 13520130=============================[]")

def logo():
    print("""\033[36m\n\n
                        (####)
                      (#######)
                    (#########)
                   (#########)
                  (#########)
                 (#########)
   __&__         (#########)
  /     \        (#########)   |\/\/\/|     /\ /\  /\              /\
 |       |      (#########)    |      |     | V  \/  \---.    .----/  \----.
 |  (o)(o)       (o)(o)(##)    |      |      \_        /       \          /
 C   .---_)    ,_C     (##)    | (o)(o)       (o)(o)  <__.   .--\ (o)(o) /__.
  | |.___|    /___,   (##)     C      _)     _C         /     \     ()     /
  |  \__/       \     (#)       | ,___|     /____,   )  \      >   (C_)   <
  /_____\        |    |         |   /         \     /----'    /___\____/___\
 /_____/ \       OOOOOO        /____\          ooooo             /|    |\
/         \     /      \      /      \        /     \           /        \
    \n\n\033[0m""")

def welcome():
    print("""\033[31m \nOnce again, welcome to CONVEX HULL SOLVER by Nelsen \033[0m""")
    print("""\033[32m This program was created by Nelsen Putra 13520130 to fulfill the second small project
of IF2211 Algorithm Strategies. \033[0m""")

def menu():
    print("""\033[33m Main Menu: \n\033[0m""")
    print("""\033[37m
    1. Start CONVEX HULL!
    2. Exit Program\n\033[0m""")
    choice1 = int(input("""\033[36m Enter menu: \033[0m """))
    if (choice1 == 1):
        print("These is the list of datasets that exist in this program:")
        print("""\033[37m
        1. Dataset iris
        2. Dataset digits
        3. Dataset wine
        4. Dataset breast_cancer\n\033[0m""")
        choice2 = int(input("""\033[36m Enter dataset: \033[0m """))
        if (choice2 == 1):
            iris()
        elif (choice2 == 2):
            digits()
```

```
        elif (choice2 == 3):
            wine()
        elif (choice2 == 4):
            breast_cancer()
    elif (choice1 == 2):
        exit()

def thankYou():
    print("""\033[36m
```



```
    \033[0m""")

def iris():
        from sklearn import datasets
        data = datasets.load_iris()
        df = pd.DataFrame(data.data, columns = data.feature_names)
        df['label'] = pd.DataFrame(data.target)
        print(df.shape)
        print(df.head)
        col1 = int(input("Enter the first column pair: "))
        col2 = int(input("Enter the second column pair: "))
        fileName = input("Enter file name: ")
        madeByPython(df, "Library Python
Spicy",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName+"python")
        madeByMe(df, "Library
myConvexHull",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName)

def digits():
        from sklearn import datasets
        data = datasets.load_digits()
        df = pd.DataFrame(data.data, columns = data.feature_names)
        df['label'] = pd.DataFrame(data.target)
        print(df.shape)
        print(df.head)
        col1 = int(input("Enter the first column pair: "))
        col2 = int(input("Enter the second column pair: "))
        fileName = input("Enter file name: ")
        madeByPython(df, "Library Python
Spicy",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName+"python")
        madeByMe(df, "Library
myConvexHull",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName)
```

```
def wine():
        from sklearn import datasets
        data = datasets.load_wine()
        df = pd.DataFrame(data.data, columns = data.feature_names)
        df['label'] = pd.DataFrame(data.target)
        print(df.shape)
        print(df.head)
        col1 = int(input("Enter the first column pair: "))
        col2 = int(input("Enter the second column pair: "))
        fileName = input("Enter file name: ")
        madeByPython(df, "Library Python
Spicy",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName+"python")
        madeByMe(df, "Library
myConvexHull",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName)

def breast_cancer():
        from sklearn import datasets
        data = datasets.load_breast_cancer()
        df = pd.DataFrame(data.data, columns = data.feature_names)
        df['label'] = pd.DataFrame(data.target)
        print(df.shape)
        print(df.head)
        col1 = int(input("Enter the first column pair: "))
        col2 = int(input("Enter the second column pair: "))
        fileName = input("Enter file name: ")
        madeByPython(df, "Library Python
Spicy",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName+"python")
        madeByMe(df, "Library
myConvexHull",data.feature_names[col1],data.feature_names[col2],col1,col2,data.target_names,fileName)

def madeByMe(df, title, xLabel, yLabel, xCol, yCol, labelName, fileName):
    plt.figure(figsize = (10, 6))
    labelSize = len(df['label'].unique())
    color = color2(labelSize)

    plt.title(title)
    plt.xlabel(xLabel)
    plt.ylabel(yLabel)

    print("Such points that are located in convex hull are in the list below:")
    for i in range(labelSize):
        bucket = df[df['label'] == i]
        bucket = bucket.iloc[:, [xCol, yCol]].values
        hull = myConvexHull(bucket)
        plt.scatter(bucket[:, 0], bucket[:, 1], label = labelName[i], color = color[i])
        print(hull)
        for simplex in hull:
            plt.plot(bucket[simplex, 0], bucket[simplex, 1], color = color[i])
```

```
    plt.legend()
    plt.savefig('output/' + fileName)
    plt.show()

def madeByPython(df, title, xLabel, yLabel,xCol, yCol, labelName, fileName):
    plt.figure(figsize = (10, 6))
    labelSize = len(df['label'].unique())
    color = color1(labelSize)

    plt.title(title)
    plt.xlabel(xLabel)
    plt.ylabel(yLabel)

    for i in range(labelSize):
        bucket = df[df['label'] == i]
        bucket = bucket.iloc[:, [xCol, yCol]].values
        hull = ConvexHull(bucket)
        plt.scatter(bucket[:, 0], bucket[:, 1], label = labelName[i], color = color[i])
        for simplex in hull.simplices:
            plt.plot(bucket[simplex, 0], bucket[simplex, 1], color = color[i])
    plt.legend()
    plt.savefig('output/' + fileName)

def color1(n):
    color = ['b','r','g','c','m','y','k','w']
    if n > len(color):
        for i in (range(n - len(color))):
            r = random.random()
            g = random.random()
            b = random.random()
            color.append((r, g, b))
    return color

def color2(n):
    color = ['y','b','m','c','g','y','k','w']
    if n > len(color):
        for i in (range(n - len(color))):
            r = random.random()
            g = random.random()
            b = random.random()
            color.append((r, g, b))
    return color

if __name__ == "__main__":
    header()
    logo()
    welcome()
    menu()
```

```
thankYou()
```

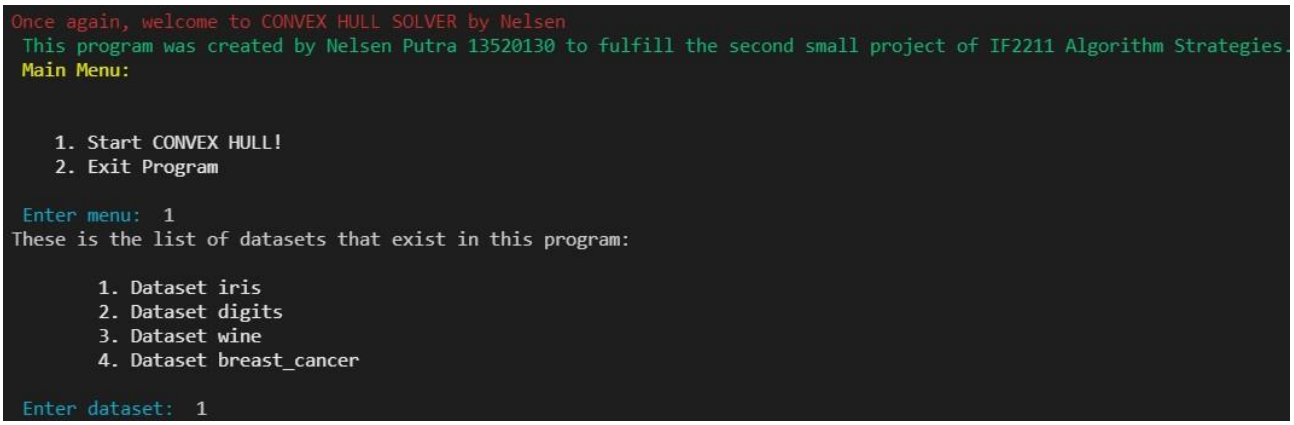
BAB III
Screenshot Input/Output Program

Eksekusi program dengan menampilkan input/output dapat dilakukan dengan 2 cara, yakni dengan menggunakan terminal pada IDE seperti VSCode dan menggunakan *file* visualization.ipynb.

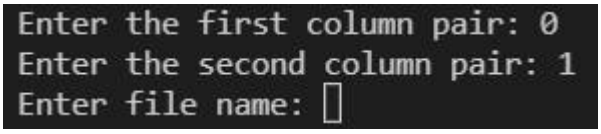
1. Eksekusi program dengan terminal VSCode
   - Tampilan awal

- Memilih menu

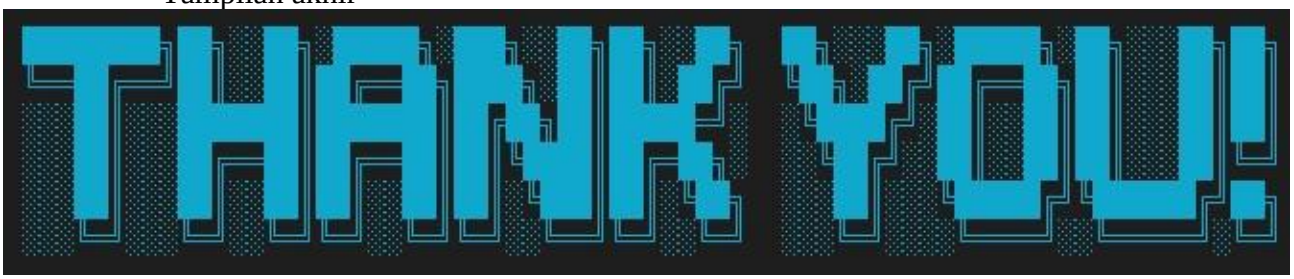
- Memilih dataset



- Memasukkan nama file



- Tampilan akhir



2. Menggunakan file visualization.ipynb yang menampilkan 4 dataset sebagai testcase yang digunakan

In [1]:
```
# Dataset 1: load_iris
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_iris()
df = pd.DataFrame(data.data, columns = data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
```

```
(150, 5)
```

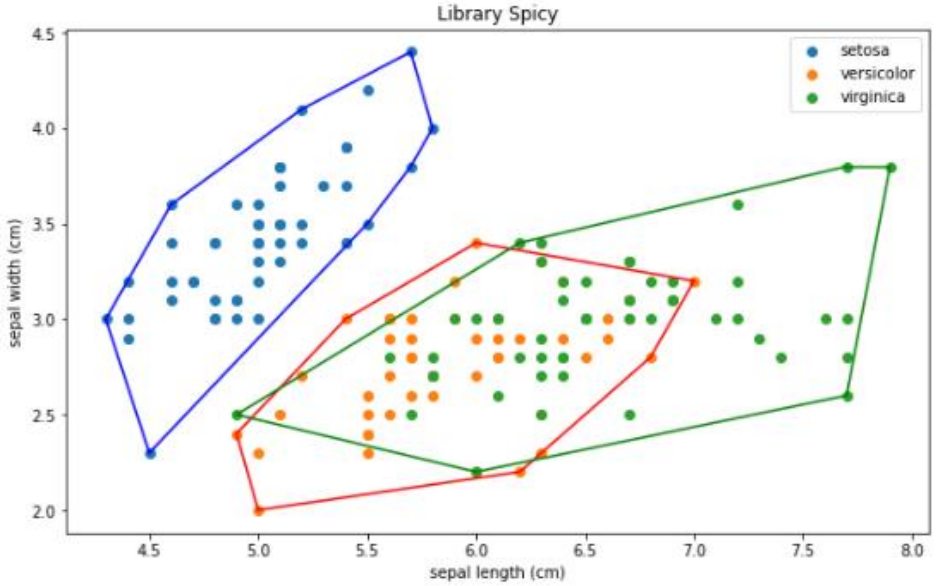
Out[1]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

In [2]:
```
# Visualisasi Convex Hull dengan Library Python
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Library Spicy')
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
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull.simplices)
    for simplex in hull.simplices:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[41 13]
 [36 41]
 [22 13]
 [22 32]
 [15 14]
 [15 32]
 [18 14]
 [18 36]]
[[35  0]
 [34  7]
 [34 35]
 [10  7]
 [10 18]
 [26  0]
 [26 18]]
[[19  6]
 [18 31]
 [18 19]
 [48  6]
 [17 31]
 [17 48]]
```

Out[2]:
```
<matplotlib.legend.Legend at 0x208ab2e7400>
```

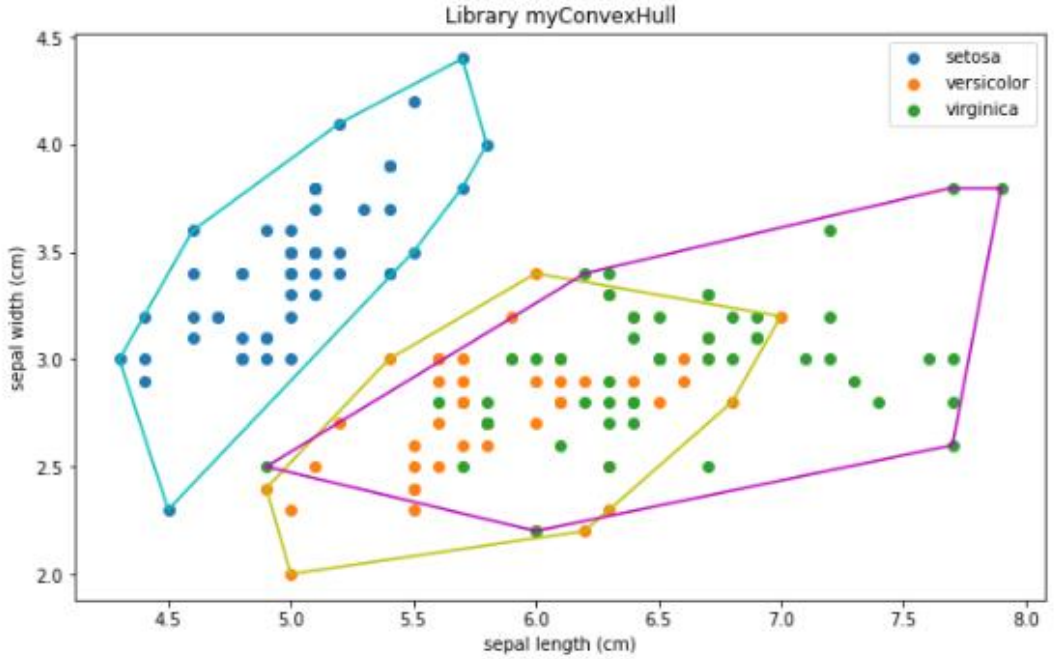


In [3]:

```
# Visualisasi Convex Hull dengan Library myConvexHull
import matplotlib.pyplot as plt
from myConvexHull import myConvexHull
plt.figure(figsize = (10, 6))
colors = ['c','y','m']
plt.title('Library myConvexHull')
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
    hull = myConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull)
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[13, 22], [22, 32], [32, 15], [15, 14], [14, 18], [18, 36], [36, 41], [41, 13]]
[[7, 34], [34, 35], [35, 0], [0, 26], [26, 18], [18, 10], [10, 7]]
[[6, 48], [48, 17], [17, 31], [31, 18], [18, 19], [19, 6]]
```

Out[3]: <matplotlib.legend.Legend at 0x208ad5ab100>

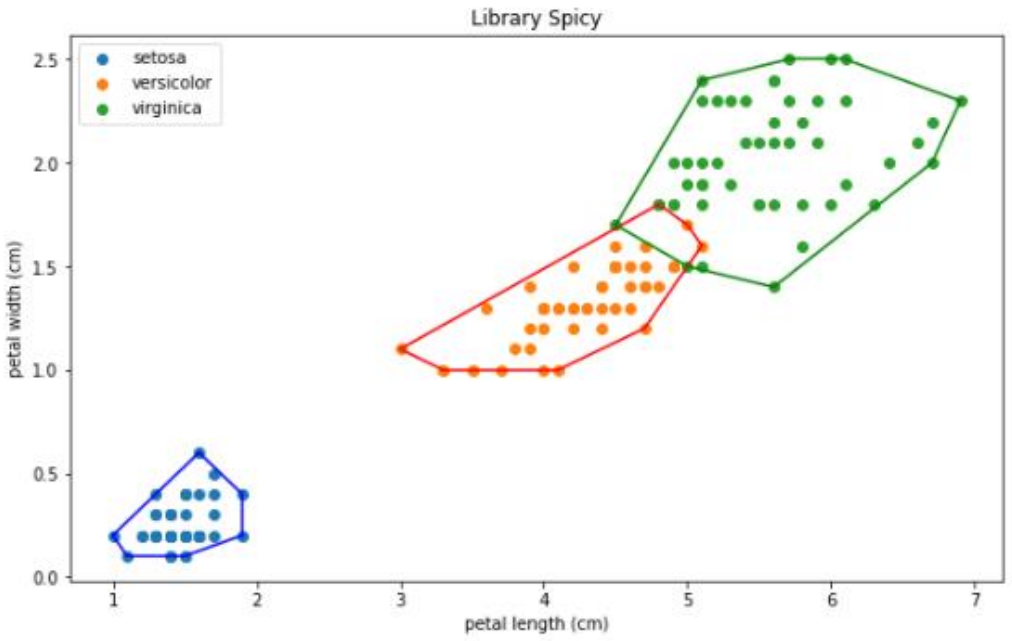


In [4]:

```
# Visualisasi Convex Hull dengan Library Python
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Library Spicy')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[3])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:, [2,3]].values
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull.simplices)
    for simplex in hull.simplices:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[43 22]
 [44 24]
 [44 43]
 [ 9 24]
 [13 22]
 [13  9]]
[[20 48]
 [27 33]
 [27 20]
 [ 7 48]
 [ 7 17]
 [23 33]
 [23 17]]
[[14  6]
 [19  6]
 [19 34]
 [22 18]
 [22 34]
 [ 9 18]
 [44 14]
 [44  9]]
```

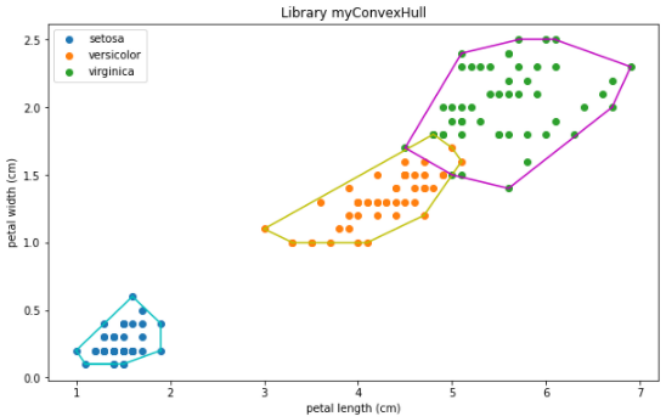
Out[4]: <matplotlib.legend.Legend at 0x208ad46c9d0>



In [5]:
```
# Visualisasi Convex Hull dengan Library myConvexHull
import matplotlib.pyplot as plt
from myConvexHull import myConvexHull
plt.figure(figsize = (10, 6))
colors = ['c','y','m']
plt.title('Library myConvexHull')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[3])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:, [2,3]].values
    hull = myConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull)
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[22, 16], [16, 43], [43, 44], [44, 24], [24, 32], [32, 9], [9, 13], [13, 12], [12, 13], [13, 12], [12, 13], [13, 22]]
[[48, 20], [20, 27], [27, 33], [33, 23], [23, 17], [17, 7], [7, 48]]
[[6, 14], [14, 44], [44, 0], [0, 9], [9, 18], [18, 22], [22, 34], [34, 19], [19, 6]]
d:\Tugas kuliah semester 4\Strategi Algoritma\tucil2_nelsen\IF2211-convexHull\src\function.py:99: RuntimeWarning: invalid value encountered in arccos
  degree = np.degrees(np.arccos(cosinusValue))
```

Out[5]:
```
<matplotlib.legend.Legend at 0x208ad5d5d60>
```



In [6]:
```
# Dataset 2: load_breast_cancer
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_breast_cancer()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
```

```
(569, 31)
```

Out[6]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	c
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	

5 rows × 31 columns

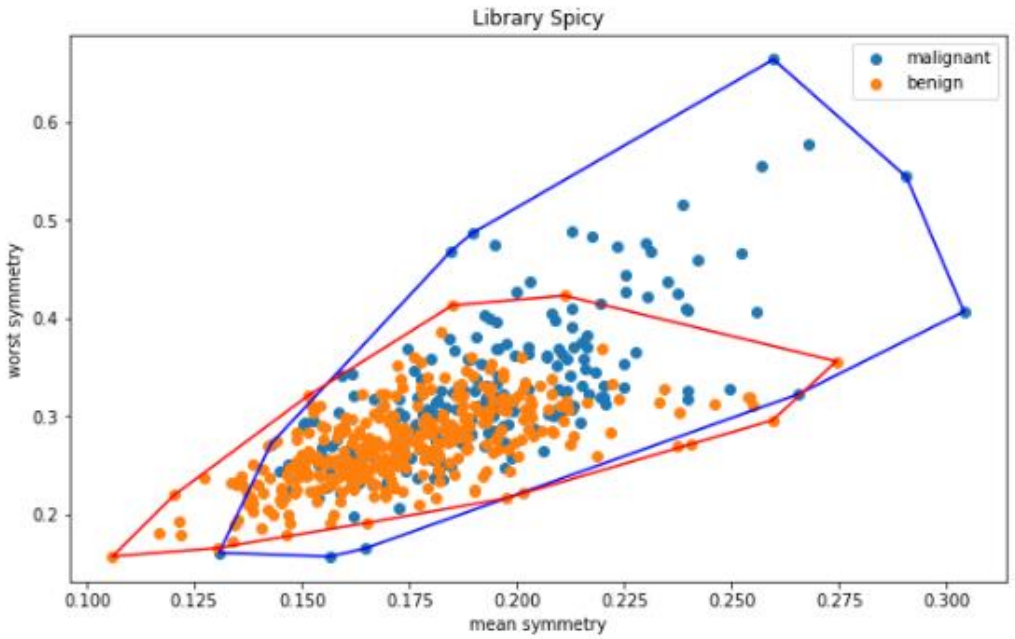
Out[6]:

mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	Target
0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0

In [7]:
```
# Visualisasi Convex Hull dengan Library Python
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Library Spicy')
plt.xlabel(data.feature_names[8])
plt.ylabel(data.feature_names[28])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:, [8,28]].values
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull.simplices)
    for simplex in hull.simplices:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[ 55  22]
 [ 55   3]
 [ 34 111]
 [ 34 110]
 [ 72  22]
 [ 72 110]
 [115 111]
 [115 191]
 [ 32   3]
 [ 32 191]]
[[ 18  13]
 [ 99  18]
 [ 48 355]
 [143  13]
 [143 325]
 [275  99]
 [ 94 325]
 [ 94  48]
 [213 355]
 [213 275]]
```

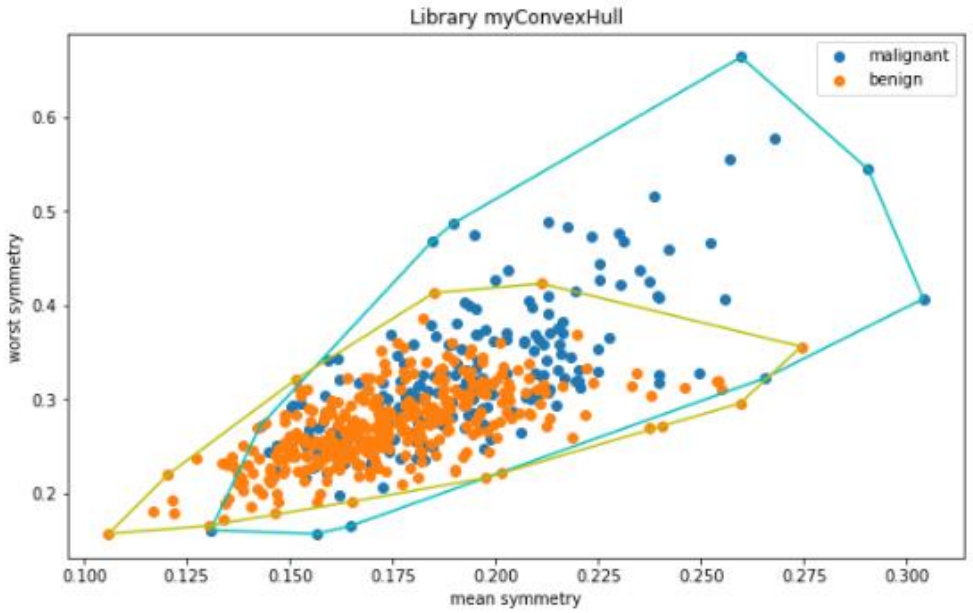
Out[7]: <matplotlib.legend.Legend at 0x208ad6e20d0>



In [8]:
```
# Visualisasi Convex Hull dengan Library myConvexHull
import matplotlib.pyplot as plt
from myConvexHull import myConvexHull
plt.figure(figsize = (10, 6))
colors = ['c','y','m']
plt.title('Library myConvexHull')
plt.xlabel(data.feature_names[8])
plt.ylabel(data.feature_names[28])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:, [8,28]].values
    hull = myConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull)
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[111, 115], [115, 191], [191, 32], [32, 3], [3, 55], [55, 22], [22, 72], [72, 110], [110, 34], [34, 111]]
[[355, 213], [213, 275], [275, 99], [99, 18], [18, 13], [13, 143], [143, 325], [325, 94], [94, 48], [48, 355]]
```

Out[8]: <matplotlib.legend.Legend at 0x208ad75cbb0>



In [9]:

```
# Dataset 3: load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_digits()
df = pd.DataFrame(data.data, columns = data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
```

```
(1797, 65)
```

Out[9]:

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6	pixel_0_7	pixel_1_0	pixel_1_1	...	pixel_6_7	pixel_7_0	pixel_7_1	pixel_7_2	pixel_7_3	pixel_7_4	pixel_7
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	6.0	13.0	10.0	
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	11.0	16.0	1
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	3.0	11.0	1
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	0.0	0.0	0.0	7.0	13.0	13.0	
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	2.0	16.0	

5 rows × 65 columns

Out[9]:

_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6	pixel_0_7	pixel_1_0	pixel_1_1	...	pixel_6_7	pixel_7_0	pixel_7_1	pixel_7_2	pixel_7_3	pixel_7_4	pixel_7_5	pixel_7_6	pixel_7_7	Target
.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	6.0	13.0	10.0	0.0	0.0	0.0	0
.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	11.0	16.0	10.0	0.0	0.0	1
.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	3.0	11.0	16.0	9.0	0.0	2
.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	0.0	0.0	0.0	7.0	13.0	13.0	9.0	0.0	0.0	3
.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	2.0	16.0	4.0	0.0	0.0	4

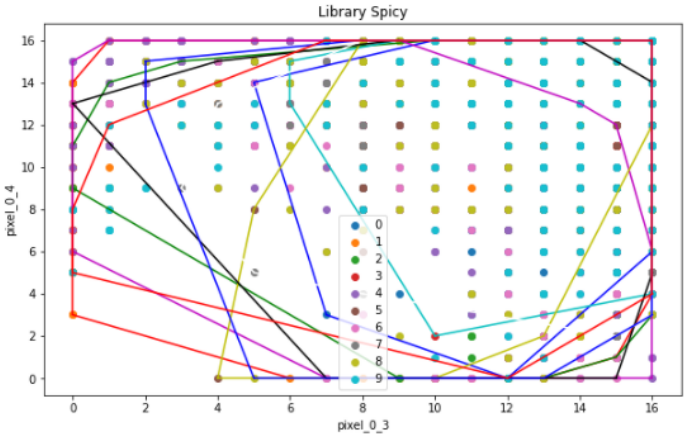
In [10]:

```
# Visualisasi Convex Hull dengan Library Python
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g','c','m','y','k','w','b','r','g','c']
plt.title('Library Spicy')
plt.xlabel(data.feature_names[3])
plt.ylabel(data.feature_names[4])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:, [3,4]].values
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull.simplices)
    for simplex in hull.simplices:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[ 98 122]
 [ 98 125]
 [141 122]
 [ 60 125]
 [ 71 141]
 [ 71  60]]
[[ 22   1]
 [ 96  22]
 [  8   1]
 [  8 178]
 [ 60  91]
 [ 60 178]
 [ 53  91]
 [ 53  96]]
[[167   7]
 [ 28  12]
 [  8   7]
 [  8   5]
 [  4  12]
 [  4   5]
 [140  86]
 [140  28]
 [ 23 167]
 [ 23  86]]
[[51 50]
 [10 51]
 [62 59]
 [62 50]
 [57 10]
 [57 59]]
[[ 33  30]
 [ 66 166]
 [142  30]
 [142 166]
 [ 64 131]
 [ 64  66]
 [  6 131]
 [128   6]
 [128  33]]
[[  0   6]
 [111   3]
 [ 12   0]
 [  4   6]
 [  4 111]
 [ 23   3]
 [ 23  12]]
[[ 71 155]
 [ 78  71]
 [ 33 155]
 [141 107]
 [131  78]
 [131 141]
 [ 57 107]
 [ 57  33]]
[[146   4]
 [ 49  68]
 [ 21   4]
 [ 21  19]
 [ 67  68]
 [ 67 146]
 [110  19]
 [110  49]]
[[112  62]
```

```
 [112  19]
 [170  19]
 [ 78  62]
 [ 78   5]
 [111   5]
 [111 170]]
[[ 72  67]
 [ 72   0]
 [130   0]
 [ 68  67]
 [ 68 155]
 [123 155]
 [123 130]]
```

Out[10]: <matplotlib.legend.Legend at 0x208ada11400>

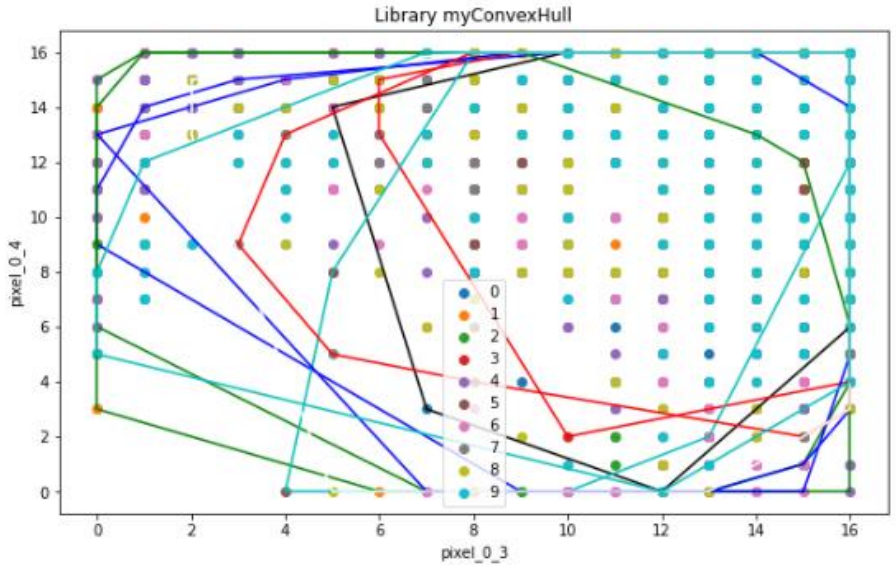


In [11]:

```
# Visualisasi Convex Hull dengan Library myConvexHull
import matplotlib.pyplot as plt
from myConvexHull import myConvexHull
plt.figure(figsize = (10, 6))
colors = ['k','g','b','r','g','c','b','r','w','c','m','y']
plt.title('Library myConvexHull')
plt.xlabel(data.feature_names[3])
plt.ylabel(data.feature_names[4])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:, [3,4]].values
    hull = myConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull)
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[122, 141], [141, 35], [35, 71], [71, 17], [17, 49], [49, 60], [60, 125], [125, 98], [98, 122]]
[[1, 8], [8, 2], [2, 40], [40, 10], [10, 68], [68, 178], [178, 37], [37, 56], [56, 60], [60, 91], [91, 53], [53, 92], [92, 96], [96, 22], [22, 1]]
[[7, 8], [8, 5], [5, 4], [4, 12], [12, 25], [25, 28], [28, 2], [2, 103], [103, 140], [140, 86], [86, 23], [23, 18], [18, 167], [167, 7]]
[[59, 57], [57, 10], [10, 3], [3, 13], [13, 22], [22, 26], [26, 51], [51, 50], [50, 62], [62, 59]]
[[2, 22], [22, 131], [131, 4], [4, 6], [6, 128], [128, 33], [33, 30], [30, 142], [142, 160], [160, 161], [161, 166], [166, 66], [66, 2]]
[[6, 4], [4, 111], [111, 3], [3, 23], [23, 12], [12, 0], [0, 6]]
[[155, 33], [33, 57], [57, 10], [10, 45], [45, 54], [54, 107], [107, 53], [53, 141], [141, 15], [15, 131], [131, 78], [78, 7], [7, 34], [34, 66], [66,
71], [71, 155]]
[[68, 67], [67, 146], [146, 4], [4, 15], [15, 18], [18, 21], [21, 19], [19, 110], [110, 49], [49, 68]]
[[62, 78], [78, 5], [5, 27], [27, 31], [31, 87], [87, 101], [101, 111], [111, 7], [7, 23], [23, 98], [98, 170], [170, 19], [19, 112], [112, 62]]
[[67, 68], [68, 155], [155, 29], [29, 26], [26, 123], [123, 5], [5, 21], [21, 11], [11, 130], [130, 0], [0, 72], [72, 67]]
```

Out[11]: <matplotlib.legend.Legend at 0x208adb04250>



In [12]:

```
# Dataset ke 4: load_wine
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_wine()
df = pd.DataFrame(data.data, columns = data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
```

```
(178, 14)
```

Out[12]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315_of_diluted_wines	pr
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1(
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1(
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	2.93	

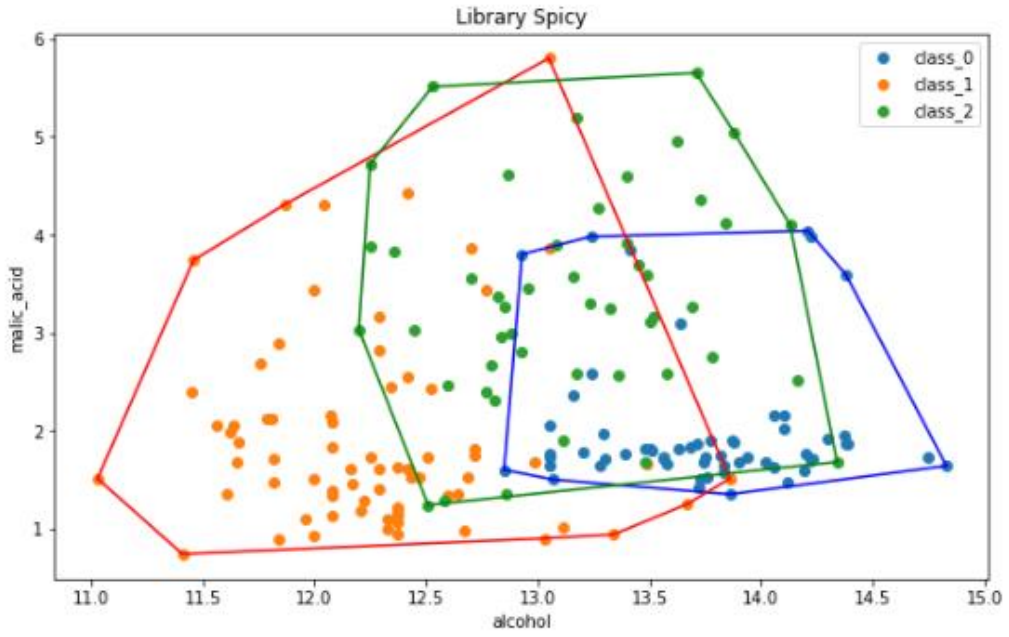
Out[12]:

malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315_of_diluted_wines	proline	Target
1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065.0	0
1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050.0	0
2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185.0	0
1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480.0	0
2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735.0	0

In [13]:
```
# Visualisasi Convex Hull dengan Library Python
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g','c','m','y','k','w','b','r','g','c']
plt.title('Library Spicy')
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
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull.simplices)
    for simplex in hull.simplices:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[21 23]
 [ 9  8]
 [46  8]
 [46 45]
 [43 45]
 [43 21]
 [38 23]
 [38  9]]
[[64 12]
 [54 56]
 [51 56]
 [65 64]
 [65 51]
 [ 3 12]
 [ 3  9]
 [17 54]
 [17  9]]
[[ 4 40]
 [ 4 28]
 [ 7 43]
 [47 28]
 [ 6 40]
 [ 6  7]
 [16 43]
 [16 47]]
```

Out[13]:
```
<matplotlib.legend.Legend at 0x208adb6c250>
```

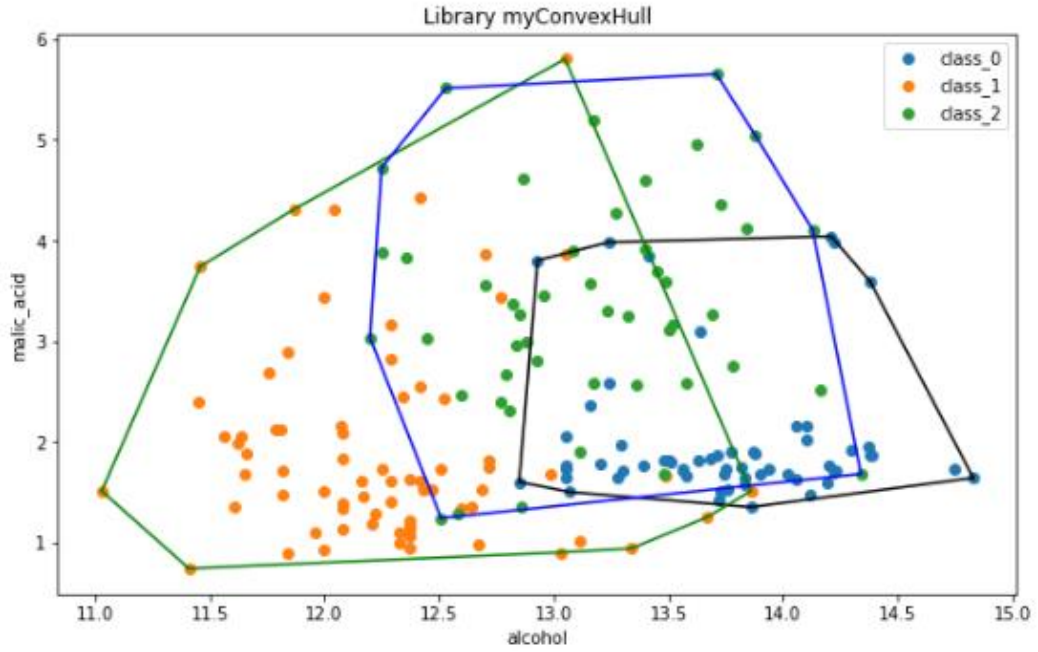


In [14]:

```
# Visualisasi Convex Hull dengan Library myConvexHull
import matplotlib.pyplot as plt
from myConvexHull import myConvexHull
plt.figure(figsize = (10, 6))
colors = ['k','g','b','r','g','c','b','r','w','c','m','y']
plt.title('Library myConvexHull')
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
    hull = myConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label = data.target_names[i])
    print(hull)
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
[[23, 21], [21, 43], [43, 45], [45, 46], [46, 8], [8, 9], [9, 38], [38, 23]]
[[56, 51], [51, 65], [65, 64], [64, 12], [12, 3], [3, 9], [9, 17], [17, 54], [54, 56]]
[[40, 6], [6, 7], [7, 43], [43, 16], [16, 47], [47, 28], [28, 4], [4, 40]]
```

Out[14]: <matplotlib.legend.Legend at 0x208adcae6d0>



BAB IV
Lampiran

No	Poin	Ya	Tidak
1	Pustaka myConvexHull berhasil dibuat dan tidak ada kesalahan	V	
2	Convex hull yang dihasilkan sudah benar	V	
3	Pustaka myConvexHull dapat digunakan untuk menampilkan convex hull setiap label dengan warna yang berbeda.	V	
4	Bonus: program dapat menerima input dan menuliskan output untuk dataset lainnya	V	

Berikut ini adalah alamat GitHub Tugas Kecil 2 IF2211 Strategi Algoritma milk Nelsen Putra (13520130): https://github.com/nelsenputra/IF2211-convexHull.