**PROGRAM DATA KELAS COURSE CODING**

**Oleh**

**Anggota Kelompok**

**10123449-Ahmad Riski**

**10123465-Attala Arrafi**

**10123461-Dzulfikar Sadid**

**10123216-Naufal Azka**

**10123457-Rizki Eskhart**

**Untuk memenuhi salah satu tugas mata kuliah  
Algoritma dan Struktur Data 1**



**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNIK DAN ILMU KOMPUTER  
UNIVERSITAS KOMPUTER INDONESIA  
BANDUNG   
2024**

DAFTAR ISI

[DAFTAR ISI ii](#_Toc159011324)

[DAFTAR GAMBAR iii](#_Toc159011325)

[DESKRIPSI PROGRAM 1](#_Toc159011326)

[1. Deskripsi Program Algoritma: Program Data Kelas Course Coding 1](#_Toc159011327)

[2. Algoritma: 1](#_Toc159011328)

[3. Struktur Data: 2](#_Toc159011329)

[4. Bahasa Pemrograman: 2](#_Toc159011330)

[5. Manfaat: 2](#_Toc159011331)

[6. Kesimpulan: 2](#_Toc159011332)

[MENU PROGRAM 3](#_Toc159011333)

[1. Program Course Coding 3](#_Toc159011334)

[ALGORITMA 4](#_Toc159011335)

[PYTHON 12](#_Toc159011336)

[SCREENSHOT PROGRAM 29](#_Toc159011337)

[KONTRIBUSI ANGGOTA KELOMPOK 35](#_Toc159011338)

DAFTAR GAMBAR

[**Gambar1. 1 Tampilan Bar Menu Pertama** 29](#_Toc159011339)

[**Gambar1. 2 Tampilan Isi Menu Pertama** 29](#_Toc159011340)

[**Gambar1. 3 Tampilan Kolom Pencarian** 30](#_Toc159011341)

[**Gambar1. 4 Tampilan Tombol Pengurutan** 30](#_Toc159011342)

[**Gambar1. 5 Tampilan Data Kelas dan Tombol Join Class** 30](#_Toc159011343)

[**Gambar1. 6 Tampilan Form Join Class** 31](#_Toc159011344)

[**Gambar1. 7 Tampilan Bar Menu Kedua** 31](#_Toc159011345)

[**Gambar1. 8 Tampilan Isi Daftar User** 31](#_Toc159011346)

[**Gambar1. 9 Tampilan Tabel Daftar User** 32](#_Toc159011347)

[**Gambar1. 10 Tampilan Menu Ketiga** 32](#_Toc159011348)

[**Gambar1. 11 Tampilan Isi Daftar User Yang Mengikuti Kelas** 33](#_Toc159011349)

[**Gambar1. 12 Tampilan Detail User Kelas** 33](#_Toc159011350)

[**Gambar1. 13 Tampilan List User Yang Mendaftar Kelas** 34](#_Toc159011351)

[**Gambar1. 14 Tampilan Detail Kelas** 34](#_Toc159011352)

DESKRIPSI PROGRAM

1. Deskripsi Program Algoritma: Program Data Kelas Course Coding

Program yang kami buat merupakan program yang terinspirasi dari platform *course online* dimana user dapat mengikuti kelas-kelas yang tersedia, terutama kelas pemrograman.

Tujuan:

Program ini bertujuan untuk mengelola data kelas course coding, termasuk:

* Data Siswa: Menyimpan informasi tentang siswa yang mengikuti course coding, seperti nama, email, dan nomor telepon.
* Data Kelas: Menyimpan informasi tentang kelas coding yang ditawarkan, seperti nama kursus, deskripsi, dan mentor.
* Data Siswa Kelas: Menyimpan informasi tentang siswa kelas coding yang telah mendaftar kelas sebelumnya.

1. Algoritma:

Program ini akan menggunakan berbagai algoritma untuk mengelola data kelas course coding, termasuk:

* Pencarian: Algoritma pencarian akan digunakan untuk menemukan data siswa, kursus, dan kelas tertentu.
* Pengurutan: Algoritma pengurutan akan digunakan untuk mengurutkan data siswa, kursus, dan kelas, berdasarkan huruf.
* Penyisipan: Algoritma penyisipan akan digunakan untuk menambahkan data siswa, kursus, dan kelas baru ke array.
* Pembaruan: Algoritma pembaruan akan digunakan untuk memperbarui data siswa, kursus, dan kelas jika ada data yang baru masuk.

1. Struktur Data:

Program ini akan menggunakan satu struktur data untuk menyimpan data kelas course coding, yaitu:

* Array: Array akan digunakan untuk menyimpan data siswa, kursus, mentor,dan kelas.

1. Bahasa Pemrograman:

Program ini kami diimplementasikan dalam 2 bahasa pemrograman, yaitu Python dan Algoritma Pseudo-code.

1. Manfaat:

Program ini akan memberikan banyak manfaat, termasuk:

* Meningkatkan efisiensi pengelolaan data kelas course coding.
* Mempermudah akses dan pencarian data.
* Meningkatkan akurasi data.
* Mempermudah pelacakan perkembangan jumlah siswa.

1. Kesimpulan:

Program data kelas course coding adalah program yang penting untuk mengelola data kelas course coding secara efektif dan efisien. Program ini akan memberikan banyak manfaat bagi penyelenggara course coding dan siswa.

MENU PROGRAM

1. Program Course Coding

Program Course Coding menyediakan berbagai menu untuk membantu Anda belajar coding dengan mudah dan menyenangkan. Berikut adalah daftar menu beserta penjelasannya:

1. **Daftar Kelas**

* Menampilkan daftar course coding yang tersedia, beserta deskripsinya.
* Anda dapat memilih course yang sesuai dengan minat.

1. **Daftar User**

Menampilkan informasi detail tentang user kelas, seperti:

* + Nama User
  + Email User
  + Nomor Hp User

1. **Daftar User Kelas** 
   * Menampilkan informasi daftar user yang terdaftar di setiap kelas
2. **Pencarian**
   * Mencari data nama kelas dan nama user yang tersedia
3. **Pengurutan**
   * Mengurutkan data kelas dan nama sesuai abjad dari A-Z
   * Mengurutkan data kelas dan nama sesuai abjad dari Z-A

ALGORITMA

Algoritma Bootcamp Online

{I.S: User memilih beberapa menu yang ditampilkan oleh sistem}

{F.S: Menampilkan konten beberapa menu yang ditampilkan oleh sistem}

Kamus (Global):

kelas : array [1..100, 1..3] of string

user: array [1..100, 1..3] of string

user\_kelas: array[1..100, 1..3] of string

is\_ascending: boolean

Procedure bubbleSortAscending(lst: array[1..100 , 1..3] of string):

    Kamus:

    n, I ,j: integer

    temp: string

    Algoritma:

    Is\_ascending <- true

    n <- lst.length

    for i <- 1 to n do

        for j <- 1 to n - i - 1 do

            if lst[j] > lst[j + 1] then

                temp <- lst[j]

                lst[j] <- lst[j + 1]

                lst[j + 1] <- temp

            endif

        endfor

    endfor

EndProcedure

Procedure bubbleSortDescending(lst: array[1..100 , 1..3] of string):

    Kamus:

    n, I ,j: integer

    temp: string

    Algoritma:

    Is\_ascending <- true

    n <- lst.length

    for i <- 1 to n do

        for j <- 1 to n - i - 1 do

            if lst[j] < lst[j + 1] then

                temp <- lst[j]

                lst[j] <- lst[j + 1]

                lst[j + 1] <- temp

            endif

        endfor

    endfor

EndProcedure

Procedure sortAndRefreshKelas(ascending):

    Algoritma:

    if is\_ascending then

        bubbleSortAscending(kelas)

    else :

        bubbleSortDescending(kelas)

    endif

    refreshKelas()

EndProcedure

Procedure sortAndRefreshUser(ascending):

    Algoritma:

    if is\_ascending then

        bubbleSortAscending(user)

    else :

        bubbleSortDescending(user)

    endif

    refreshUser()

EndProcedure

Function binarySearchAscending(lst: array[1..100, 1..3] of string, target:string)->int

    Kamus:

    left,right,mid : integer

    Algoritma:

    left <- 0

    right <- lst.length

    while left <= right do

        mid <- (left + right) div 2

        if lst[mid][0] < target then

            left <- mid + 1

        else if target < lst[mid][0] then

            right <-  mid - 1

        else

            return mid

        endif

    endwhile

    return -1

EndFunction

Function binarySearchDescending(lst: array[1..100, 1..3] of string, target:string)->int

    Kamus:

    left,right,mid : integer

    Algoritma:

    left <- 0

    right <- lst.length

    while left <= right do

        mid <- (left + right) div 2

        if lst[mid][0] > target then

            left <- mid + 1

        else if target > lst[mid][0] then

            right <-  mid - 1

        else

            return mid

        endif

    endwhile

    return -1

EndFunction

Procedure searchKelas(query:string)

    Kamus:

    result : integer

    Algoritma:

    if is\_acending then

        bubbleSortAscending (kelas)

        result <- binarySearchAscending(kelas, query)

    else

        bubbleSortDescending (kelas)

        result <- binarySearchDescending(kelas, query)

    endif

    if result >= 0 then

        output("Found, Kelas ditemukan pada posisi " + result)

    else

        output("Not Found, Kelas tidak ditemukan")

    endif

EndProcedure

Procedure searchUser(query:string)

    Kamus:

    result : Integer

    Algoritma:

    if is\_acending then

        bubbleSortAscending(user)

        result <- binarySearchAscending(user, query)

    else

        bubbleSortDescending(user)

        result <- binarySearchDescending(user, query)

    endif

    if result >= 0 then

        output("Found, User ditemukan pada posisi " + result)

    else

        output("Not Found, User tidak ditemukan")

    endif

EndProcedure

Procedure refreshUser()

    Algoritma:

    output(user)

EndProcedure

Procedure refreshKelas()

    Algoritma:

    output(kelas)

EndProcedure

Procedure refreshUserKelas()

    Algoritma:

    output(user\_kelas)

EndProcedure

Procedure showJoinClassForm(kelas : string)

    Kamus:

    nama, email, phone : string

    Algoritma:

    Input(nama)

    Input(email)

    Input(phone)

    user[user.length+1] <- [nama, email, kelas]

    user\_kelas[user\_kelas.length+1] <- [nama, kelas]

EndProcedure

Algoritma:

kelas[1] <- ["Data Science", "Jane Smith", 3]

kelas[2] <- ["Web Development", "Alex Johnson", 5]

kelas[3] <- ["Machine Learning", "Emily Brown", 2]

kelas[4] <- ["Android Development", "Michael Wilson", 3]

kelas[5] <- ["IOS Development", "Sarah Thompson", 4]

kelas[6] <- ["UI/UX Design", "David Miller", 3]

kelas[7] <- ["Backend Development", "Jessica Davis", 5]

kelas[8] <- ["React Native", "John Doe", 4]

user[1] <- ['Dzulfikar Sadid', 'dzulfikar@mail.com','08123456789']

user[2] <- ['Ahmad Riski', 'ahmad@mail.com', '08123456789']

user[3] <- ['Naufal Azka', 'naufal@mail.com', '08123456789']

user\_kelas[1] <- ['Dzulfikar Sadid', 'Data Science']

user\_kelas[2] <- ['Ahmad Riski', 'Web Development']

user\_kelas[3] <- ['Naufal Azka', 'Machine Learning']

user\_kelas[4] <- ['Dzulfikar Sadid', 'React Native']

user\_kelas[5] <- ['Ahmad Riski', 'Backend Development']

user\_kelas[6] <- ['Naufal Azka', 'Backend Development']

sortAndRefreshKelas(true)

sortAndRefreshUser(true)

output("Selamat Datang di Bootcamp Online")

output("Silahkan pilih menu yang tersedia")

output("1. Lihat Kelas")

output("2. Lihat User")

output("3. Cari Kelas")

output("4. Cari User")

output("5. Keluar")

*input*(menu)

if menu == 1 then

show\_kelas()

elseif menu == 2 then

show\_user()

elseif menu == 3 then

*input*(query)

search\_kelas(query)

elseif menu == 4 then

*input*(query)

search\_user(query)

else

output("Terima kasih telah menggunakan Bootcamp Online")

endif

PYTHON

import tkinter as tk

from tkinter import ttk, messagebox, Toplevel

# List kelas yang tersedia

# Nama kelas, nama mentor, durasi (bulan)

kelas = [

["Data Science", "Jane Smith", 3],

["Web Development", "Alex Johnson", 5],

["Machine Learning", "Emily Brown", 2],

["Android Development", "Michael Wilson", 3],

["IOS Development", "Sarah Thompson", 4],

["UI/UX Design", "David Miller", 3],

["Backend Development", "Jessica Davis", 5],

["React Native", "John Doe", 4],

]

# List user yang terdaftar

# Nama, email, nomor telepon

user = [

['Dzulfikar Sadid', 'dzulfikar@mail.com',

'08123456789'],

['Ahmad Riski', 'ahmad@mail.com', '08123456789'],

['Naufal Azka', 'naufal@mail.com', '08123456789'],

]

# List user yang terdaftar di kelas

# Nama, kelas

user\_kelas = [

['Dzulfikar Sadid', 'Data Science'],

['Ahmad Riski', 'Web Development'],

['Naufal Azka', 'Machine Learning'],

['Dzulfikar Sadid', 'React Native'],

['Ahmad Riski', 'Backend Development'],

['Naufal Azka', 'Backend Development'],

]

# List terurut ascending atau tidak

is\_ascending = True

# Inisialisai GUI

root = tk.Tk()

root.title("Bootcamp Online")

root.geometry("1280x720")

notebook = ttk.Notebook(root)

notebook.pack(fill='both', expand=True)

kelas\_page = ttk.Frame(notebook)

user\_page = ttk.Frame(notebook)

user\_kelas\_page = ttk.Frame(notebook)

# Sorting

def bubble\_sort\_ascending(lst):

global is\_ascending

is\_ascending = True

n = len(lst)

for i in range(n - 1):

for j in range(n - i - 1):

if lst[j] > lst[j + 1]:

lst[j], lst[j + 1] = lst[j + 1], lst[j]

def bubble\_sort\_descending(lst):

global is\_ascending

is\_ascending = False

n = len(lst)

for i in range(n - 1):

for j in range(n - i - 1):

if lst[j] < lst[j + 1]:

lst[j], lst[j + 1] = lst[j + 1], lst[j]

def sort\_and\_refresh\_kelas(ascending):

if ascending:

bubble\_sort\_ascending(kelas)

else:

bubble\_sort\_descending(kelas)

refresh\_kelas()

def sort\_and\_refresh\_user(ascending):

if ascending:

bubble\_sort\_ascending(user)

else:

bubble\_sort\_descending(user)

refresh\_user()

# Searching

def binary\_search\_ascending(lst, target):

left = 0

right = len(lst) - 1

while left <= right:

mid = (left + right) // 2

if lst[mid][0] < target:

left = mid + 1

elif target < lst[mid][0]:

right = mid - 1

else:

return mid

return -1

def binary\_search\_descending(lst, target):

left = 0

right = len(lst) - 1

while left <= right:

mid = (left + right) // 2

if lst[mid][0] > target:

left = mid + 1

elif target > lst[mid][0]:

right = mid - 1

else:

return mid

return -1

def search\_kelas(query):

global is\_ascending

if is\_ascending:

bubble\_sort\_ascending(kelas)

result = binary\_search\_ascending(kelas, query)

else:

bubble\_sort\_descending(kelas)

result = binary\_search\_descending(kelas, query)

if result >= 0:

show\_kelas\_search(result + 1, kelas[result])

else:

messagebox.showinfo("Not Found",

"Kelas tidak ditemukan")

def search\_user(query):

global is\_ascending

if is\_ascending:

bubble\_sort\_ascending(user)

result = binary\_search\_ascending(user, query)

else:

bubble\_sort\_descending(user)

result = binary\_search\_descending(user, query)

if result >= 0:

show\_user\_search(result + 1, user[result])

else:

messagebox.showinfo("Not Found",

"User tidak ditemukan")

# Tampilan Halaman Kelas

def refresh\_kelas():

for widget in kelas\_page.winfo\_children():

widget.destroy()

label = tk.Label(kelas\_page, text="Daftar Kelas",

font=("Arial", 18))

label.pack(pady=10)

search\_frame = tk.Frame(kelas\_page)

search\_frame.pack(pady=5)

search\_label = tk.Label(search\_frame, text="Search:",

font=("Arial", 14))

search\_label.pack(side="left")

search\_entry = tk.Entry(search\_frame,

font=("Arial", 14))

search\_entry.pack(side="left", padx=5)

search\_button = tk.Button(search\_frame, text="Search",

font=("Arial", 14),

relief="groove", padx=5,

pady=2, width=10,

borderwidth=2, bd=1,

bg="gray", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: search\_kelas(

search\_entry.get()))

search\_button.pack(side="left", padx=5)

for i in range(len(kelas)):

kelas\_frame = tk.Frame(kelas\_page)

kelas\_frame.pack()

kelas\_button = tk.Button(kelas\_frame,

text="{}. {}".format(i + 1,

kelas[

i][

0]),

anchor="w",

font=("Arial", 14),

relief="groove", padx=10,

pady=5, width=20,

borderwidth=4, bd=1,

bg="white", fg="black",

highlightthickness=0,

highlightcolor="white",

command=lambda

kelas\_detail=kelas[

i]: show\_kelas\_detail(

kelas\_detail))

kelas\_button.pack(side="left", pady=5)

join\_button = tk.Button(kelas\_frame,

text="Join Class",

font=("Arial", 16),

relief="groove", padx=5,

pady=2, width=10,

borderwidth=2, bd=1,

bg="blue", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: show\_join\_class\_form(

kelas[i]))

join\_button.pack(side="left", pady=5)

sort\_asc\_button = tk.Button(kelas\_page,

text="Sort Ascending",

font=("Arial", 18),

relief="groove", padx=5,

pady=2, width=15,

borderwidth=2, bd=1,

bg="green", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: sort\_and\_refresh\_kelas(

True))

sort\_asc\_button.pack(pady=5)

sort\_desc\_button = tk.Button(kelas\_page,

text="Sort Descending",

font=("Arial", 18),

relief="groove", padx=5,

pady=2, width=15,

borderwidth=2, bd=1,

bg="red", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: sort\_and\_refresh\_kelas(

False))

sort\_desc\_button.pack(pady=5)

# Tampilan Halaman User

def refresh\_user():

for widget in user\_page.winfo\_children():

widget.destroy()

label = tk.Label(user\_page, text="Daftar User",

font=("Arial", 24))

label.pack(pady=10)

search\_frame = tk.Frame(user\_page)

search\_frame.pack(pady=5)

search\_label = tk.Label(search\_frame, text="Search:",

font=("Arial", 14))

search\_label.pack(side="left")

search\_entry = tk.Entry(search\_frame,

font=("Arial", 14))

search\_entry.pack(side="left", padx=5)

search\_button = tk.Button(search\_frame, text="Search",

font=("Arial", 14),

relief="groove", padx=5,

pady=2, width=10,

borderwidth=2, bd=1,

bg="gray", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: search\_user(

search\_entry.get()))

search\_button.pack(side="left", padx=5)

user\_table = ttk.Treeview(user\_page)

user\_table["columns"] = ("Nama", "Email", "Nomor")

user\_table.column("#0", width=50)

user\_table.column("Nama", width=150)

user\_table.column("Email", width=200)

user\_table.column("Nomor", width=150)

user\_table.heading("#0", text="No.")

user\_table.heading("Nama", text="Nama")

user\_table.heading("Email", text="Email")

user\_table.heading("Nomor", text="Nomor")

for i, user\_detail in enumerate(user):

user\_table.insert(parent="", index="end", iid=i,

text=i + 1, values=(

user\_detail[0], user\_detail[1], user\_detail[2]))

user\_table.pack(pady=10)

sort\_asc\_button = tk.Button(user\_page,

text="Sort Ascending",

font=("Arial", 18),

relief="groove", padx=5,

pady=2, width=15,

borderwidth=2, bd=1,

bg="green", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: sort\_and\_refresh\_user(

True))

sort\_asc\_button.pack(pady=5)

sort\_desc\_button = tk.Button(user\_page,

text="Sort Descending",

font=("Arial", 18),

relief="groove", padx=5,

pady=2, width=15,

borderwidth=2, bd=1,

bg="red", fg="white",

highlightthickness=0,

highlightcolor="white",

command=lambda: sort\_and\_refresh\_user(

False))

sort\_desc\_button.pack(pady=5)

# Tampilan Halaman User Kelas

def refresh\_user\_kelas():

global kelas

for widget in user\_kelas\_page.winfo\_children():

widget.destroy()

label = tk.Label(user\_kelas\_page, text="Daftar Kelas",

font=("Arial", 18))

label.pack(pady=10)

for i in range(len(kelas)):

kelas\_frame = tk.Frame(user\_kelas\_page)

kelas\_frame.pack()

kelas\_button = tk.Button(kelas\_frame,

text="{}. {}".format(i + 1,

kelas[

i][

0]),

anchor="w",

font=("Arial", 14),

relief="groove", padx=10,

pady=5, width=20,

borderwidth=4, bd=1,

bg="white", fg="black",

highlightthickness=0,

highlightcolor="white",

command=lambda

kelas\_detail=kelas[

i]: show\_kelas\_detail(

kelas\_detail))

kelas\_button.pack(side="left", pady=5)

join\_button = tk.Button(kelas\_frame,

text="Show Users",

font=("Arial", 16),

relief="groove", padx=5,

pady=2, width=10,

borderwidth=2, bd=1,

bg="yellow", fg="black",

highlightthickness=0,

highlightcolor="white",

command=lambda

kelas\_new=kelas[i][

0]: show\_user\_kelas(

kelas\_new))

join\_button.pack(side="left", pady=5)

# Form untuk mengikuti kelas

def show\_join\_class\_form(kelas):

form\_window = Toplevel(kelas\_page)

form\_window.title(f"Join {kelas[0]}")

# Create form labels

name\_label = tk.Label(form\_window, text="Nama Anda:")

name\_label.grid(row=1, column=0, padx=10, pady=5,

sticky="w")

name\_entry = tk.Entry(form\_window)

name\_entry.grid(row=1, column=1, padx=10, pady=5)

email\_label = tk.Label(form\_window, text="Email Anda:")

email\_label.grid(row=2, column=0, padx=10, pady=5,

sticky="w")

email\_entry = tk.Entry(form\_window)

email\_entry.grid(row=2, column=1, padx=10, pady=5)

phone\_label = tk.Label(form\_window,

text="Nomor Telephone:")

phone\_label.grid(row=3, column=0, padx=10, pady=5,

sticky="w")

phone\_entry = tk.Entry(form\_window)

phone\_entry.grid(row=3, column=1, padx=10, pady=5)

submit\_button = tk.Button(form\_window, text="Submit",

command=lambda: submit\_form(

name\_entry.get(),

email\_entry.get(),

phone\_entry.get(),

kelas[0]))

submit\_button.grid(row=4, column=0, columnspan=2,

pady=10)

# Submit form

def submit\_form(name, email, phone, kelas):

user.append([name, email, phone])

user\_kelas.append([name, kelas])

messagebox.showinfo("Berhasil",

"Selamat anda telah berhasil mendafar ke kelas ini!")

sort\_and\_refresh\_user(True)

# Details

def show\_kelas\_detail(kelas\_detail):

messagebox.showinfo("Kelas Detail",

f"Nama Kelas: {kelas\_detail[0]}\nMentor: {kelas\_detail[1]}\nDurasi: {kelas\_detail[2]} bulan")

def show\_kelas\_search(index, kelas\_detail):

messagebox.showinfo(

f"Kelas Ditemukan di Posisi {index}",

f"Nama Kelas: {kelas\_detail[0]}\nMentor: {kelas\_detail[1]}\nDurasi: {kelas\_detail[2]} bulan")

def show\_user\_detail(user\_detail):

messagebox.showinfo("User Detail",

f"Nama: {user\_detail[0]}\nEmail: {user\_detail[1]}\nNomor: {user\_detail[2]}")

def show\_user\_search(index, user\_detail):

messagebox.showinfo(f"User Ditemukan di Posisi {index}",

f"Nama: {user\_detail[0]}\nEmail: {user\_detail[1]}\nNomor: {user\_detail[2]}")

def show\_user\_kelas(kelas):

filtered\_users = []

for user in user\_kelas:

if user[1] == kelas:

filtered\_users.append(user[0])

user\_list = "\n".join(

["{}. {}".format(i + 1, user) for i, user in

enumerate(filtered\_users)])

if len(filtered\_users) == 0:

messagebox.showinfo("Filtered Users",

"Tidak ada user yang terdaftar di kelas ini")

else:

messagebox.showinfo("Filtered Users", user\_list)

# Fungsi tambahan untuk menampilkan halaman

def show\_page(notebook, page):

notebook.select(page)

def tampilkan\_menu\_gui():

notebook.add(kelas\_page, text='Daftar Kelas')

notebook.add(user\_page, text='Daftar User')

notebook.add(user\_kelas\_page, text='Daftar User Kelas')

refresh\_kelas()

refresh\_user()

refresh\_user\_kelas()

root.mainloop()

# Memanggil fungsi / Bagian Utama Program

sort\_and\_refresh\_kelas(True)

sort\_and\_refresh\_user(True)

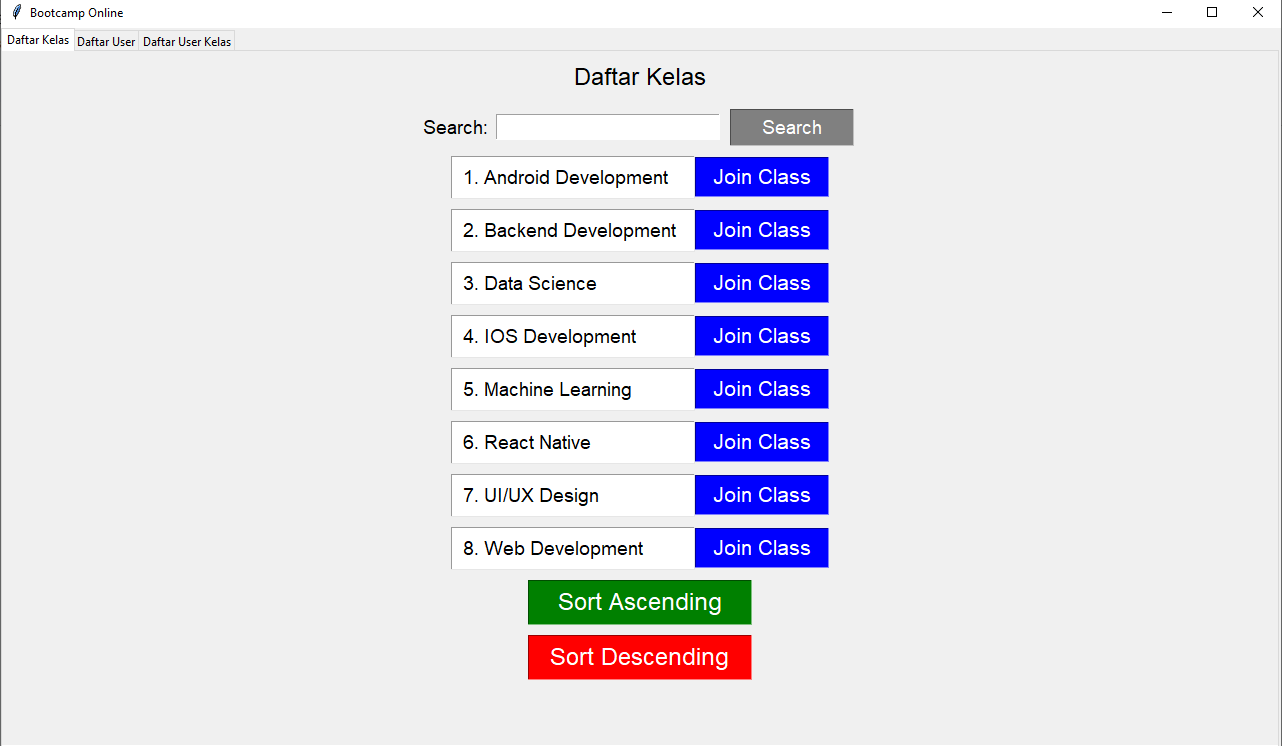
tampilkan\_menu\_gui()

SCREENSHOT PROGRAM



**Gambar1. 1  
 Tampilan Bar Menu Pertama**

Gambar menu yang muncul pertama saat program dijalankan, Ketika ditekan menu Daftar Kelas maka akan muncul **Gambar 1.2.**



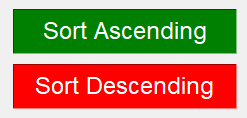
**Gambar1. 2  
 Tampilan Isi Menu Pertama**

Tampilan menu awal saat program dijalankan, terlihat ada menu Join Class untuk memasukkan data ke kelas.



**Gambar1. 3  
 Tampilan Kolom Pencarian**

Menu pencarian untuk memudahkan user dalam mencari data kelas di tampilan awal, user dapat memasukkan data nama kelas untuk menemukan kelas yang diinginkan.



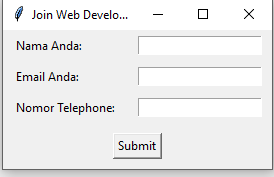
**Gambar1. 4  
 Tampilan Tombol Pengurutan**

Menu pengurutan untuk memudahkan user dalam mengurutkan data nama kelas, baik dari A-Z maupun dari Z-A.



**Gambar1. 5  
 Tampilan Data Kelas dan Tombol Join Class**

Tampilan daftar nama kelas, jika user menekan tombol Join Class maka akan muncul tampilan seperti **Gambar 1.6.**



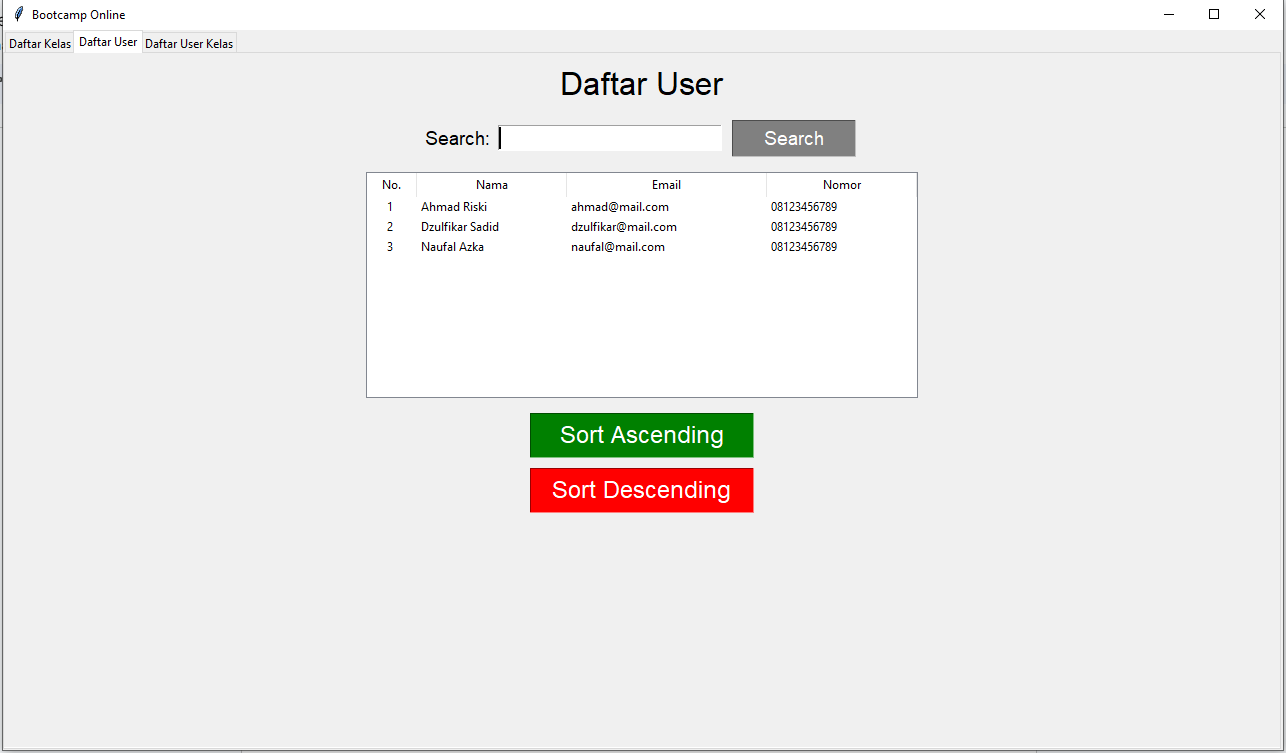
**Gambar1. 6  
 Tampilan Form Join Class**

Menu inputan untuk user, user dapat memasukkan data dirinya pada menu ini dan dapat menekan submit jika sudah selesai.



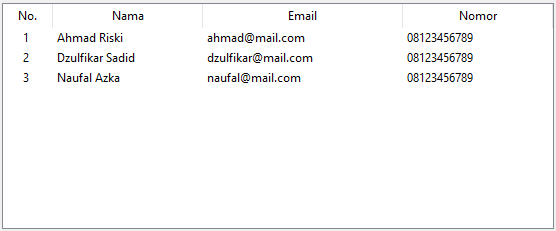
**Gambar1. 7  
 Tampilan Bar Menu Kedua**

Menu kedua, yaitu menu Daftar User, jika ditekan maka akan muncul tampilan seperti **Gambar 1.8.**



**Gambar1. 8  
 Tampilan Isi Daftar User**

Tampilan menu Daftar User, disini pun terdapat fitur pencarian dan pengurutan untuk memudahkan user.



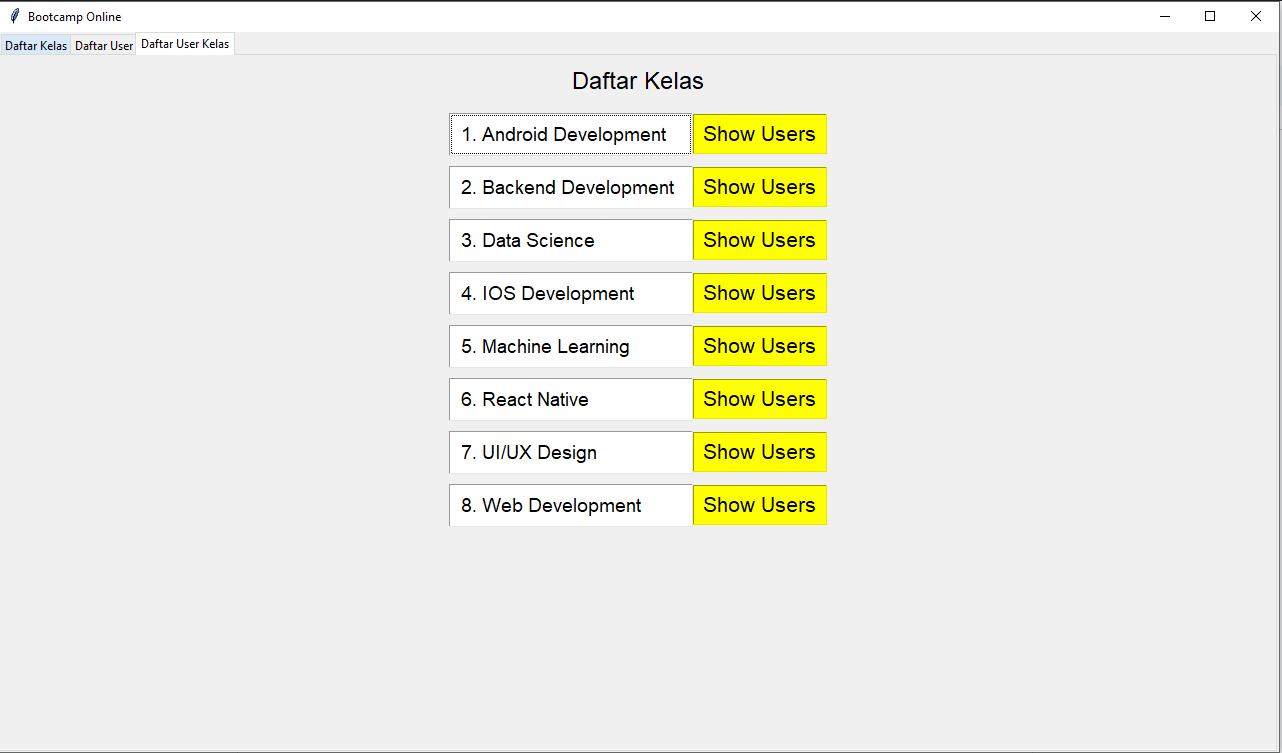
**Gambar1. 9  
 Tampilan Tabel Daftar User**

User dapat melihat list user yang terdaftar di kelas.



**Gambar1. 10  
 Tampilan Menu Ketiga**

Menu ketiga, yaitu Daftar User Kelas, jika ditekan, maka akan muncul tampilan seperti **Gambar 1.11.**



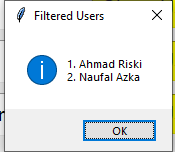
**Gambar1. 11  
 Tampilan Isi Daftar User Yang Mengikuti Kelas**

Tampilan menu Daftar User Kelas.



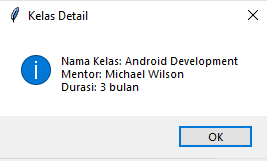
**Gambar1. 12  
 Tampilan Detail User Kelas**

Tampilan daftar kelas yang tersedia, user dapat melihat list nama yang terdaftar di dalam kelas, seperti pada **Gambar 1.13.**



**Gambar1. 13  
 Tampilan List User Yang Mendaftar Kelas**

Tampilan list nama yang terdaftar dikelas.



**Gambar1. 14  
 Tampilan Detail Kelas**

Tampilan Detail Kelas, jika user menekan nama kelas, baik di menu pertama, maupun menu ketiga.

KONTRIBUSI ANGGOTA KELOMPOK

1. Ahmad Riski = Riset ide, sistem dan membuat pseudo-code
2. Attala Arrafi = Membantu pseudo-code dan menyusun daftar isi dan daftar gambar
3. Dzulfikar Sadid = Mengubah pseudo-code menjadi python
4. Naufal Azka = Menyusun laporan dan melengkapi fitur
5. Rizki Eskhart = Menyusun laporan dan mengembangkan ide