LAPORAN
PEMBUATAN APLIKASI COUNTMATRIX UNTUK PERHITUNGAN MATRIKS SEDERHANA

Disusun dan digunakan untuk memenuhi

Tugas Besar Praktikum Pemrograman 1

Dosen Pengampu :
Arif Amrulloh, S.Kom., M.Kom

Disusun Oleh :

1. Aditya Prabu Mukhti (2211104037)
2. Hamid Khaeruman (2211104040)
3. Maulidya Fatima Marsyanni (2211104041)
4. Rafli Fauzan Ra'uf (2211104050)
5. Dimas Cahyo Margono (2211104060)

S1SE-06-B

PROGRAM STUDI REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2023

KATA PENGANTAR

Puji dan syukur dipanjatkan hanya kepada Allah, Tuhan Yang Maha Esa yang telah melimpahkan kemampuan dan kekuatan untuk menyelesaikan laporan ini.

Laporan ini dibuat untuk memenuhi syarat Tugas Besar dari mata kuliah Praktikum Pemrograman 1. Produk yang dibuat ini bertujuan menambah keterampilan serta pengalaman untuk meningkatkan kualitas sebagai lulusan Institut Teknologi Telkom Purwokerto, sehingga nantinya kami memiliki kompetensi yang mampu bersaing dengan kompetitor lain di dunia kerja. Pembuatan laporan ini juga nantinya berguna bagi adik kelas sebagai media pembelajaran praktikum mahasiswa.

Tidak lupa kami ucapkan terima kasih kepada Kepala Program Studi S1 Rekayasa Perangkat Lunak, dosen pengampu mata kuliah Praktikum Pemrograman 1, Asisten Praktikum, serta rekan-rekan seluruhnya.

Demikian laporan kami susun, mohon maaf apabila dalam penulisan masih terdapat kesalahan dan kekurangan dalam laporan ini. Semoga laporan yang disusun ini dapat bermanfaat untuk Institut Teknologi Telkom Purwokerto dan bagi di lingkungan masyarakat.

Purwokerto, 1 Juli 2023

Penulis

DAFTAR ISI

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Python merupakan bahasa pemrograman yang terkenal dan serba guna yang digunakan dalam berbagai aplikasi, termasuk pemrosesan data, pembelajaran mesin, dan pengembangan web. Python juga memiliki banyak perpustakaan yang dapat mempermudah pengembangan aplikasi. Salah satu contoh aplikasi yang dapat dikembangkan menggunakan Python adalah perhitungan matriks. Matriks adalah struktur data yang digunakan untuk merepresentasikan data dalam bentuk tabel dua dimensi. Perhitungan matriks ini memiliki beragam kegunaan dalam berbagai aplikasi, seperti pemrosesan citra, pengolahan sinyal, dan pembelajaran mesin.

Aplikasi perhitungan matriks bernama "CountMatrix" ini memiliki manfaat yang sangat besar dalam pengembangan ilmu pengetahuan dan teknologi. Dalam bidang matematika, aplikasi ini dapat digunakan untuk menyelesaikan berbagai masalah yang melibatkan perhitungan matriks, seperti sistem persamaan linear, transformasi linier, dan analisis data. Selain itu, aplikasi ini juga dapat digunakan dalam bidang sains, seperti fisika, kimia, dan biologi, untuk menyelesaikan berbagai masalah yang melibatkan perhitungan matriks.

Dalam pembuatan aplikasi CountMatrix dengan bahasa pemrograman Python, terdapat beberapa konsep dasar yang perlu dipahami, seperti variabel, tipe data, dan operator. Selain itu, juga perlu dipahami konsep matriks, seperti definisi matriks, operasi matriks, dan determinan matriks. Dengan memahami konsep-konsep dasar tersebut, pembuatan aplikasi tersebut dapat dilakukan dengan lebih mudah dan efektif.

Dalam laporan ini, akan dijelaskan secara detail tentang konsep-konsep dasar yang digunakan dalam pembuatan aplikasi tersebut, serta langkah-langkah yang dilakukan dalam pembuatan aplikasi tersebut. Maka, laporan ini diharapkan dapat memberikan pemahaman yang lebih baik tentang

aplikasi perhitungan matriks serta manfaatnya dalam pengembangan ilmu pengetahuan dan teknologi.

1.2 Tujuan

1. Membuat program yang dapat melakukan perhitungan matriks.
2. Menambah pengetahuan dalam ilmu koding dasar.
3. Mengetahui cara pembuatan aplikasi dengan memanfaatkan fungsi-fungsi tertentu.
4. Mengetahui kelebihan dan kekurangan dari aplikasi CountMatrix.
5. Menambah wawasan mahasiswa dan menerapkan hasil dari pembahasan materi.

1.3 Manfaat

Aplikasi CountMatrix memiliki manfaat yang sangat besar dalam pengembangan ilmu pengetahuan dan teknologi. Beberapa manfaat dari pembuatan aplikasi ini adalah sebagai berikut:

1. Menyelesaikan berbagai masalah matematika: Aplikasi perhitungan matriks dapat digunakan untuk menyelesaikan berbagai masalah matriks, seperti penjumlahan, perkalian, transpose, dan determinan. Dalam bidang matematika, aplikasi ini sangat berguna untuk mempermudah perhitungan dan analisis data.
2. Meningkatkan kemampuan analitis dan pemecahan masalah: Dengan menguasai bahasa pemrograman Python dan mengimplementasikannya dalam aplikasi perhitungan matriks, seseorang dapat meningkatkan kemampuan analitis dan pemecahan masalah. Hal ini sangat berguna dalam berbagai bidang, seperti sains, teknologi, dan bisnis.
3. Mempermudah pengembangan ilmu pengetahuan dan teknologi: Aplikasi CountMatrix dapat mempermudah pengembangan ilmu pengetahuan dan teknologi. Dalam bidang sains, aplikasi ini dapat digunakan untuk menyelesaikan berbagai masalah yang melibatkan perhitungan matriks, seperti dalam fisika, kimia, dan biologi.

BAB II
ANALISIS KERJA APLIKASI

Aplikasi CountMatrix yang dibuat menggunakan bahasa pemrograman Python dapat divisualisasikan dengan menggunakan matriks sebagai objek. Dalam Python, matriks dapat dibuat dengan menggunakan array nested list. Penggunaan array adalah metode pembuatan matriks yang lebih efisien dan mudah digunakan untuk melakukan operasi pada matriks. Beberapa operasi yang dapat dilakukan pada matriks adalah penjumlahan, perkalian, transpose, dan determinan. Selain itu, untuk mengevaluasi kinerja model klasifikasi, dapat digunakan metode function untuk mengelompokkan elemen dan ordo dari matriks yang ditentukan.

Dalam pembuatan aplikasi tersebut, perlu dilakukan analisis terhadap kebutuhan pengguna dan desain aplikasi. Analisis kebutuhan pengguna dapat dilakukan dengan mengidentifikasi fitur-fitur yang dibutuhkan oleh pengguna, seperti jenis operasi matriks yang ingin dilakukan dan format input/output yang diinginkan. Selain itu, fitur dari aplikasi juga perlu dipertimbangkan dengan baik agar aplikasi dapat berjalan dengan efisien dan efektif. Desain aplikasi meliputi pemilihan struktur data yang tepat, algoritma yang efisien, dan alur prosedur yang mudah digunakan.

Setelah melakukan analisis kebutuhan pengguna dan desain aplikasi, langkah selanjutnya adalah melakukan perancangan aplikasi. Perancangan ini dapat dilakukan dengan menggunakan bahasa pemrograman Python dan memanfaatkan berbagai metode, seperti input/output, array, looping, sorting, dan if/else. Selama proses ini, perlu dilakukan pengujian aplikasi untuk memastikan bahwa aplikasi berjalan dengan baik dan sesuai dengan kebutuhan pengguna. Pengujian aplikasi dapat dilakukan dengan menggunakan teknik pengujian fungsional dan pengujian non-fungsional.

Setelah aplikasi selesai dirancang dan diuji, langkah selanjutnya adalah melakukan dokumentasi dan pemeliharaan aplikasi. Dokumentasi aplikasi dapat berupa dokumentasi kode, dokumentasi pengguna, dan dokumentasi teknis. Pemeliharaan aplikasi meliputi perbaikan bug, peningkatan fitur, dan peningkatan

performa aplikasi. Pemeliharaan aplikasi perlu dilakukan secara berkala untuk memastikan bahwa aplikasi selalu berjalan dengan baik dan sesuai dengan kebutuhan pengguna.

BAB III
PEMBAHASAN

3.1 Input-Output dan Implementasinya

Input adalah data atau sesuatu yang kita masukkan ke dalam program untuk di proses. Process adalah tahapan yang harus dilakukan oleh program yang akan menghasilkan output. Output adalah informasi atau data yang dihasilkan setelah dilakukan pemrosesan.

Berikut adalah contoh implementasinya pada program CountMatrix :

Source code :

```
rows = int(input("Masukkan jumlah baris matriks: "))
cols = int(input("Masukkan jumlah kolom matriks: "))
```

Untuk mencetak data dari input yang dimasukkan user, tambahkan syntax print(rows) dan print(cols) di bawahnya.

3.2 If / Else dan Implementasinya

Statement if/else pada pemrograman adalah salah satu bentuk struktur kontrol yang digunakan untuk mengambil keputusan berdasarkan kondisi tertentu. Pada dasarnya, statement if/else memungkinkan program untuk menjalankan blok kode tertentu jika suatu kondisi terpenuhi, dan menjalankan blok kode lain jika kondisi tersebut tidak terpenuhi.

Berikut adalah contoh implementasinya pada program CountMatrix :

Source code :

```
result = []
  if operation == 4:  # Perkalian dengan skalar
      matrix_choice = input("Pilih matriks yang akan dikalikan (A untuk matriks A, B untuk matriks B): ")
      scalar = float(input(f"Masukkan skalar untuk matriks {matrix_choice}: "))
      if matrix_choice == 'A':
          matrix = matrix_A
      elif matrix_choice == 'B':
          matrix = matrix_B
      else:
          print("Pilihan matriks tidak valid.")
          return result

      # Eksekusi Fungsi Perkalian Skalar
      for i in range(len(matrix)):
          row = []
          for j in range(len(matrix[0])):
              element = matrix[i][j] * scalar
              row.append(element)
          result.append(row)
  else:
      for i in range(len(matrix_A)):
          row = []
          for j in range(len(matrix_A[0])):
              if operation == 1:  # Penjumlahan
                  element = matrix_A[i][j] + matrix_B[i][j]
              elif operation == 2:  # Pengurangan
                  element = matrix_A[i][j] - matrix_B[i][j]
              elif operation == 3:  # Perkalian
                  element = sum(matrix_A[i][k] * matrix_B[k][j] for k in range(len(matrix_B)))
              row.append(element)
          result.append(row)
  return result
```

Pada source code tersebut, statement if/else digunakan untuk menentukan kondisi pemilihan operasi dari matriks yang akan dihitung.

3.3 Looping dan Implementasinya

Looping statement pada pemrograman merujuk pada konstruksi yang memungkinkan pengulangan atau iterasi dari serangkaian pernyataan atau blok kode. Dalam pemrograman, loop digunakan untuk menjalankan satu atau lebih pernyataan berulang kali sampai kondisi tertentu terpenuhi. Hal ini memungkinkan pengguna untuk mengotomatisasi tugas-tugas yang berulang dan mengolah data dalam jumlah besar dengan efisien.

Berikut adalah contoh implementasinya pada program CountMatrix :

Source code :

```
for i in range(len(matrix)):
    row = []
    for j in range(len(matrix[0])):
        element = matrix[i][j] * scalar
        row.append(element)
    result.append(row)
```

Pada source code tersebut, dilakukan pengulangan untuk mengalikan sebuah matriks X dengan besaran skalar tertentu, yang mana untuk meletakkan hasilnya menerapkan statement for loop.

3.4 Array dan Implementasinya

Dalam pemrograman, array merupakan struktur data yang digunakan untuk menyimpan kumpulan elemen dengan tipe data yang sama. Array memungkinkan kita untuk menyimpan beberapa nilai dalam satu variabel dan mengaksesnya menggunakan indeks.

Pengimplementasian array dimulai dengan mendeklarasikan tipe data elemen array dan menginisialisasi array dengan ukuran tertentu. Di banyak bahasa pemrograman, langkah-langkah ini dapat dilakukan sebagai berikut:

1. Deklarasikan tipe data elemen array. Misalnya, jika Anda ingin membuat array bilangan bulat, Anda akan mendeklarasikan tipe data sebagai "nama_array = [elemen_a, elemen_b, elemen_c]".

2. Tentukan ukuran array. Ini adalah jumlah elemen yang dapat disimpan dalam array. Misalnya, untuk array dengan 5 elemen, ukurannya akan ditentukan sebagai 5.
3. Inisialisasikan array. Di beberapa bahasa pemrograman, array secara otomatis diinisialisasi dengan nilai default sesuai tipe datanya. Misalnya, array bilangan bulat akan diinisialisasi dengan nilai 0. Namun, Anda juga dapat menginisialisasi array dengan nilai khusus jika diinginkan.

Berikut adalah contoh implementasinya pada program CountMatrix :

Source code :

```
def perform_operation(matrix_A, matrix_B, operation):
    result = []
    if operation == 4:  # Perkalian dengan skalar
        matrix_choice = input("Pilih matriks yang akan dikalikan (A untuk matriks A, B untuk matriks B): ")
        scalar = float(input(f"Masukkan skalar untuk matriks {matrix_choice}: "))
        if matrix_choice == 'A':
            matrix = matrix_A
        elif matrix_choice == 'B':
            matrix = matrix_B
        else:
            print("Pilihan matriks tidak valid.")
            return result
```

Pada source code tersebut, terdapat array **result** yang bersifat nullable agar bisa diisi dengan elemen yang fleksibel untuk menampungnya menjadi susunan elemen matriks.

3.5 Function dan Implementasinya

Statement function dalam pemrograman merujuk pada sebuah blok kode yang memiliki tugas tertentu dan dapat dipanggil atau dieksekusi oleh program utama. Fungsi ini digunakan untuk mengorganisir dan memisahkan logika program menjadi bagian-bagian yang lebih kecil dan terstruktur. Dengan menggunakan statement function, kita dapat menghindari duplikasi kode, meningkatkan keterbacaan, dan memudahkan pemeliharaan kode.

Berikut adalah contoh implementasinya pada program CountMatrix :

Source code :

```
def input_matrix(rows, cols):
    matrix = []
    print("Masukkan elemen matriks:")
    for i in range(rows):
        row = [float(input(f"Masukkan elemen matriks pada baris {i+1}, kolom {j+1}: ")) for j in
range(cols)]
        matrix.append(row)
    return matrix
```

Pada source code tersebut, terdapat fungsi **input_matrix** yang menampung berbagai syntax untuk menginputkan elemen-elemen pada matriks X pada posisi baris dan kolom tertentu.

3.6 Sorting dan Implementasinya

Sorting statement dalam pemrograman merujuk pada proses mengurutkan elemen-elemen dalam suatu struktur data berdasarkan aturan tertentu. Tujuan dari pengurutan adalah untuk mengatur elemen-elemen tersebut menjadi urutan yang teratur, sehingga memudahkan akses dan pencarian data.

Implementasi sorting pada Python dapat dilakukan dengan menggunakan berbagai algoritma sorting yang telah tersedia. Beberapa algoritma sorting yang umum digunakan di Python antara lain:

1. Bubble Sort: Algoritma ini bekerja dengan membandingkan pasangan elemen secara berurutan dan menukar posisi jika diperlukan. Proses ini berulang hingga seluruh elemen terurut dengan benar. Bubble sort memiliki kompleksitas waktu $O(n^2)$, di mana n adalah jumlah elemen yang akan diurutkan.
2. Insertion Sort: Algoritma ini bekerja dengan membagi daftar menjadi dua bagian, yaitu bagian terurut dan bagian belum terurut. Pada setiap iterasi, satu elemen dari bagian belum terurut dipilih dan dimasukkan ke posisi yang tepat di dalam bagian terurut. Proses ini berulang hingga seluruh elemen terurut dengan benar. Insertion sort juga memiliki kompleksitas waktu $O(n^2)$.
3. Merge Sort: Algoritma ini menggunakan pendekatan divide and conquer untuk mengurutkan elemen-elemen dalam daftar. Prosesnya melibatkan pembagian daftar menjadi dua bagian yang lebih kecil, mengurutkan masing-masing bagian secara rekursif, dan kemudian menggabungkan kembali kedua bagian tersebut menjadi satu daftar terurut. Merge sort memiliki kompleksitas waktu $O(n \log n)$, di mana n adalah jumlah elemen yang akan diurutkan.

Berikut adalah contoh implementasinya pada program CountMatrix :

Source code :

```
def sort_matrix(matrix, ascending=True):
    flattened_matrix = [element for row in matrix for element in row]
    flattened_matrix.sort(reverse=not ascending)
    sorted_matrix = []
    index = 0
    for _ in range(len(matrix)):
        row = []
        for _ in range(len(matrix[0])):
            row.append(flattened_matrix[index])
            index += 1
        sorted_matrix.append(row)
    return sorted_matrix
```

Pada source code tersebut, terdapat fungsi **sort_matrix** yang menampung berbagai syntax untuk mengurutkan elemen-elemen pada hasil matriks yang telah dihitung secara ascending maupun descending tergantung dari pemanggilan fungsi tersebut.

Untuk hasil lengkapnya dapat dilihat pada link GitHub berikut :

GitHub - HamidKhaeruman/CountMatrix

BAB IV
TIMELINE PENGERJAAN

Penmbuatan aplikasi CountMatrix ini berlangsung selama tiga minggu sebagaimana tertera pada tabel berikut.

No.	Jenis Kegiatan	Minggu			Penanggung Jawab
		1	2	3	
1	Penentuan Topik Program				Seluruh Anggota
2	Pembuatan Alur Program				Rafli Fauzan Rauf
3	Penyusunan Komponen Program				Dimas Cahyo Margono
4	Klasifikasi Fungsi Program				Hamid Khaeruman
5	Klasifikasi Kondisi Program				Aditya Prabu Mukhti
6	Pembuatan Input dan Output Program				Maulidya Fatima Marsyanni
7	Pengujian Program				Seluruh Anggota
8	Debugging dan Penambahan Fitur				Dimas Cahyo Margono
9	Pembuatan Laporan Akhir				Seluruh Anggota

BAB V
PENUTUP

5.1. Kesimpulan

Selama pelaksanaan mata kuliah Praktikum Pemrograman 1, kami mendapatkan banyak pengetahuan baru. Berikut adalah hasil yang diperoleh mahasiswa setelah melaksanakan mata kuliah tersebut :

1. Mendapatkan ilmu baru, yaitu membuat aplikasi yang dapat membantu dalam perhitungan matriks dalam konsep yang sederhana.
2. Kami dapat lebih bertanggung jawab, meningkatkan kedisiplinan, dan ketelitian dalam menyelesaikan tugas (dalam hal ini adalah pembuatan aplikasi CountMatrix).
3. Kami dapat memahami dan mampu mengimplementasikan aplikasi CountMatrix dalam kehidupan sehari-hari.
4. Kami memahami cara berkoordinasi dengan tim selama merancang aplikasi CountMatrix.

5.2. Saran

Saran untuk dosen pengampu mata kuliah dan asisten praktikum setelah pelaksanaan mata kuliah Praktikum Pemrograman 1 antara lain :

1. Sebaiknya asisten praktikum lebih memperhatikan format pembuatan tugas, terutama tugas besar.
2. Karena mata kuliah dilaksanakan secara luring, maka alangkah baiknya untuk mempersiapkan secara matang materi-materi yang diimplementasikan pada setiap pertemua. Hal ini bertujuan agar kami tidak bingung dalam memahami materi tersebut.
3. Sebaiknya asisten praktikum tidak memberitahukan informasi secara tiba-tiba terkait pengunduran maupun percepatan jadwal pelaksanaan mata kuliah.