Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Tugas Besar II IF2211 Strategi Algoritma
Semester II Tahun 2020/2021

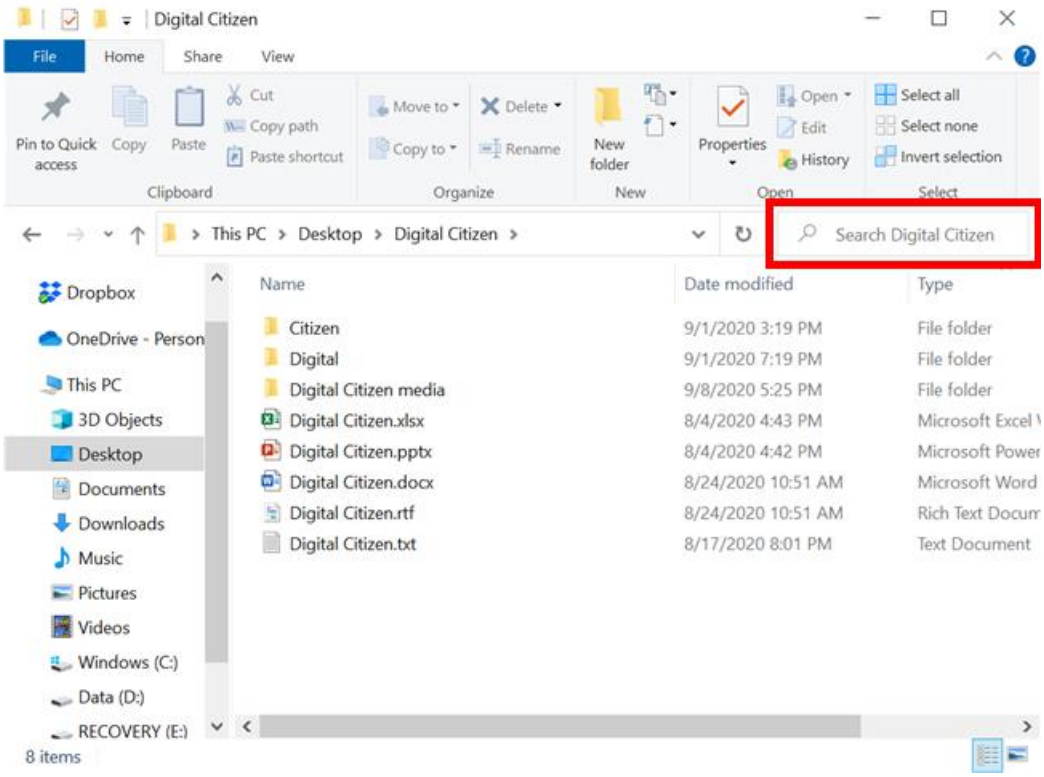
Pengaplikasian Algoritma BFS dan DFS dalam Implementasi *Folder Crawling*

Batas pengumpulan : 25 Maret 2022 pukul 23.59 WIB
Arsip pengumpulan :

- *Source* program yang bisa dijalankan disertai *readme.txt*
- Laporan (*soft copy*)

Latar belakang :

Pada saat kita ingin mencari file spesifik yang tersimpan pada komputer kita, seringkali task tersebut membutuhkan waktu yang lama apabila kita melakukannya secara manual. Bukan saja harus membuka beberapa folder hingga dapat mencapai directory yang diinginkan, kita bahkan dapat lupa di mana kita meletakkan file tersebut. Sebagai akibatnya, kita harus membuka berbagai folder secara satu persatu hingga kita menemukan file yang diinginkan. Hal ini pastinya akan sangat memakan waktu dan energi.



Gambar 1. Fitur Search pada Windows 10 File Explorer
(Sumber: https://www.digitalcitizen.life/wp-content/uploads/2020/10/explorer_search_10.png)

Meskipun demikian, kita tidak perlu cemas dalam menghadapi persoalan tersebut sekarang. Pasalnya, hampir seluruh sistem operasi sudah menyediakan fitur *search* yang dapat digunakan untuk mencari file yang kita inginkan. Kita cukup memasukkan *query* atau kata kunci pada kotak pencarian, dan komputer akan mencarikan seluruh file pada suatu *starting directory* (hingga seluruh *children*-nya) yang berkorespondensi terhadap *query* yang kita masukkan.

Fitur ini diimplementasikan dengan teknik *folder crawling*, di mana mesin komputer akan mulai mencari file yang sesuai dengan *query* mulai dari *starting directory* hingga seluruh *children* dari *starting directory* tersebut sampai satu file pertama/seluruh file ditemukan atau tidak ada file yang ditemukan. Algoritma yang dapat dipilih untuk melakukan *crawling* tersebut pun dapat bermacam-macam dan setiap algoritma akan memiliki teknik dan konsekuensinya sendiri. Oleh karena itu, penting agar komputer memilih algoritma yang tepat sehingga hasil yang diinginkan dapat ditemukan dalam waktu yang singkat.

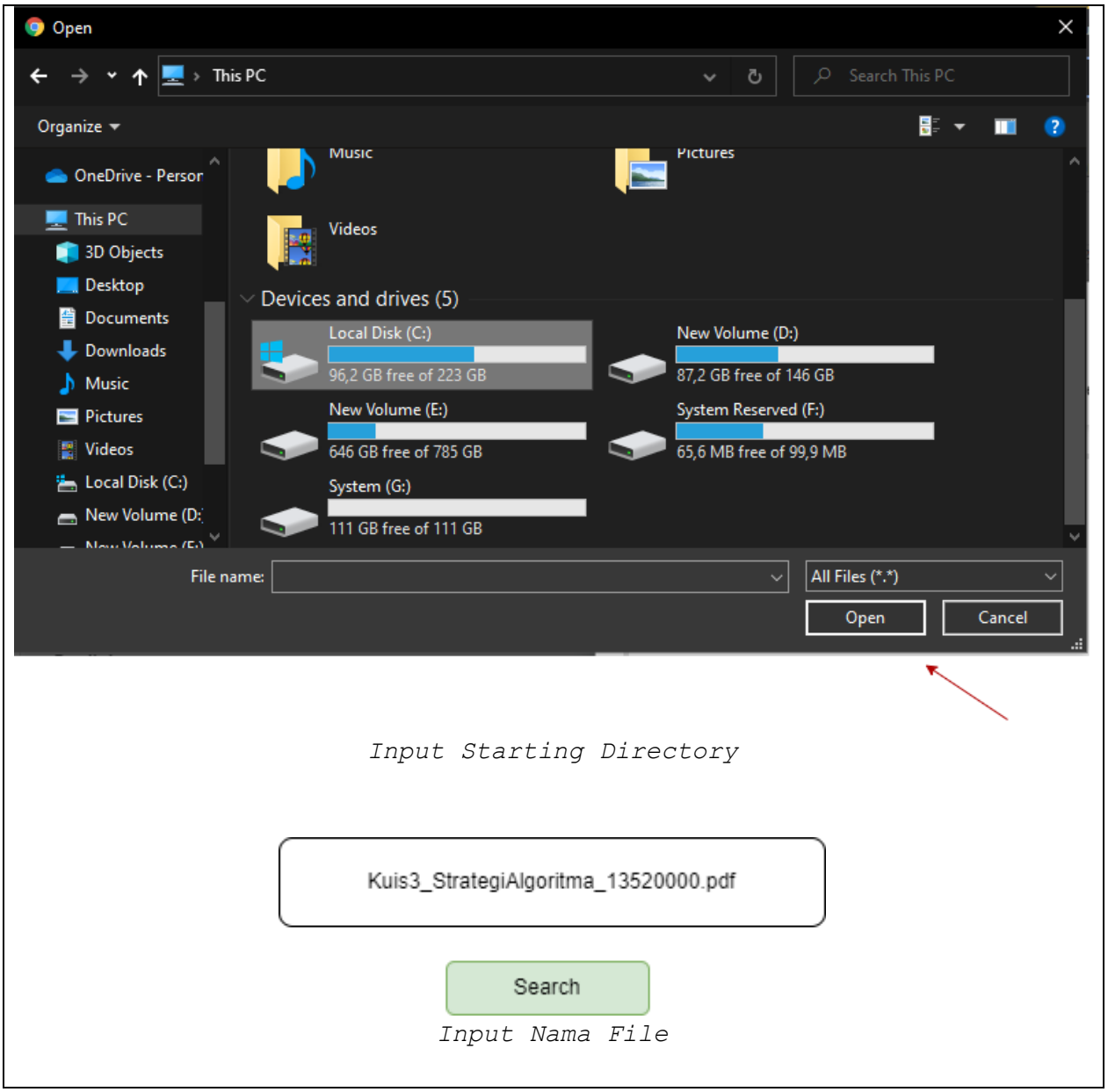
Deskripsi tugas:

Dalam tugas besar ini, Anda akan diminta untuk membangun sebuah aplikasi GUI sederhana yang dapat memodelkan fitur dari *file explorer* pada sistem operasi, yang pada tugas ini disebut dengan *Folder Crawling*. Dengan memanfaatkan algoritma *Breadth First Search* (BFS) dan *Depth First Search* (DFS), Anda dapat menelusuri folder-folder yang ada pada direktori untuk mendapatkan direktori yang Anda inginkan. Anda juga diminta untuk memvisualisasikan hasil dari pencarian *folder* tersebut dalam bentuk pohon.

Selain pohon, Anda diminta juga menampilkan list *path* dari daun-daun yang bersesuaian dengan hasil pencarian. *Path* tersebut diharuskan memiliki *hyperlink* menuju folder *parent* dari file yang dicari, agar file langsung dapat diakses melalui *browser* atau *file explorer*. Contoh hal-hal yang dimaksud akan dijelaskan di bawah ini.

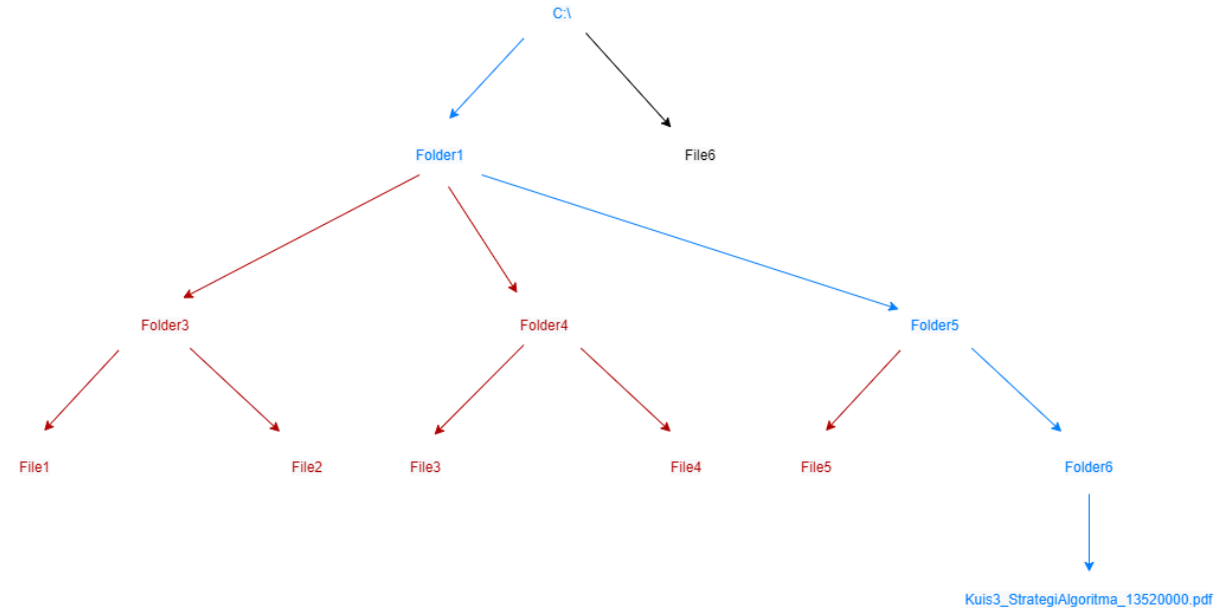
Contoh Input dan Output Program

Contoh masukan aplikasi:



Gambar 2. Contoh input program

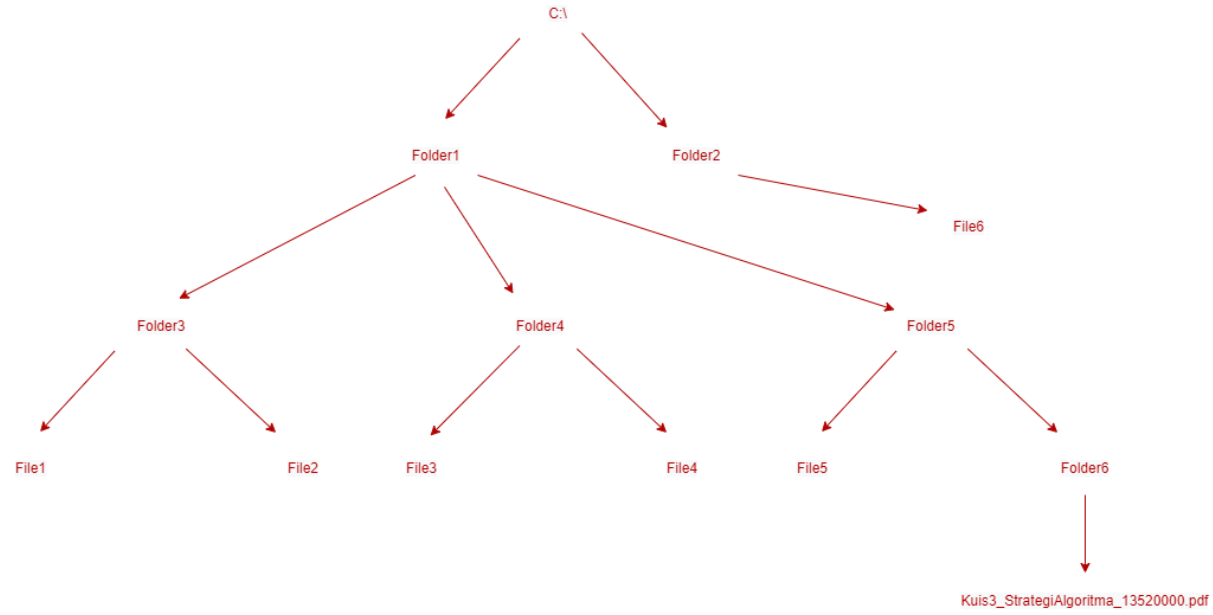
Contoh output aplikasi:



List Path:
1. C:\Folder1\Folder5\Folder6\Kuis3_StrategiAlgoritma_13520000.pdf

Gambar 3. Contoh output program

Misalnya pengguna ingin mengetahui langkah *folder crawling* untuk menemukan file Kuis3_StrategiAlgoritma_13520000.pdf.
Maka, path pencarian DFS adalah sebagai berikut. C:\ → Folder1 → Folder3 → File1 → Folder3 → File2 → Folder3 → Folder1 → Folder4 → File3 → Folder4 → File4 → Folder4 → Folder1 → Folder5 → File5 → Folder5 → Folder6 → Kuis3_StrategiAlgoritma_13520000.pdf.
Pada gambar di atas, rute yang dilewati pada pencarian DFS diwarnai dengan warna merah. Sedangkan, rute untuk menuju tempat file berada diberi warna biru. Rute yang masuk antrian tapi belum diperiksa diberi warna hitam. Anda bebas menentukan warnanya asalkan dibedakan antara ketiga hal tersebut.

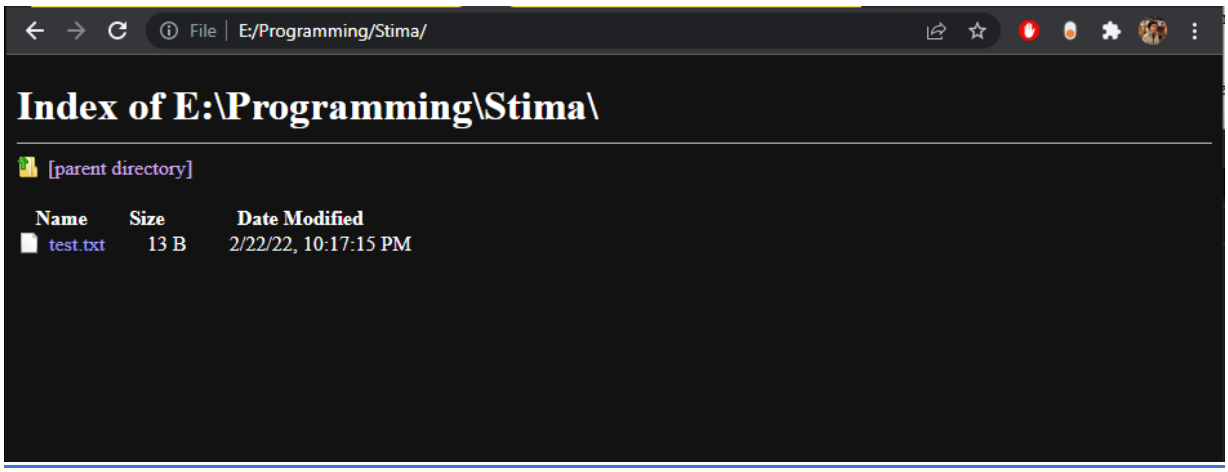


List Path:
Tidak ada path yang sesuai.

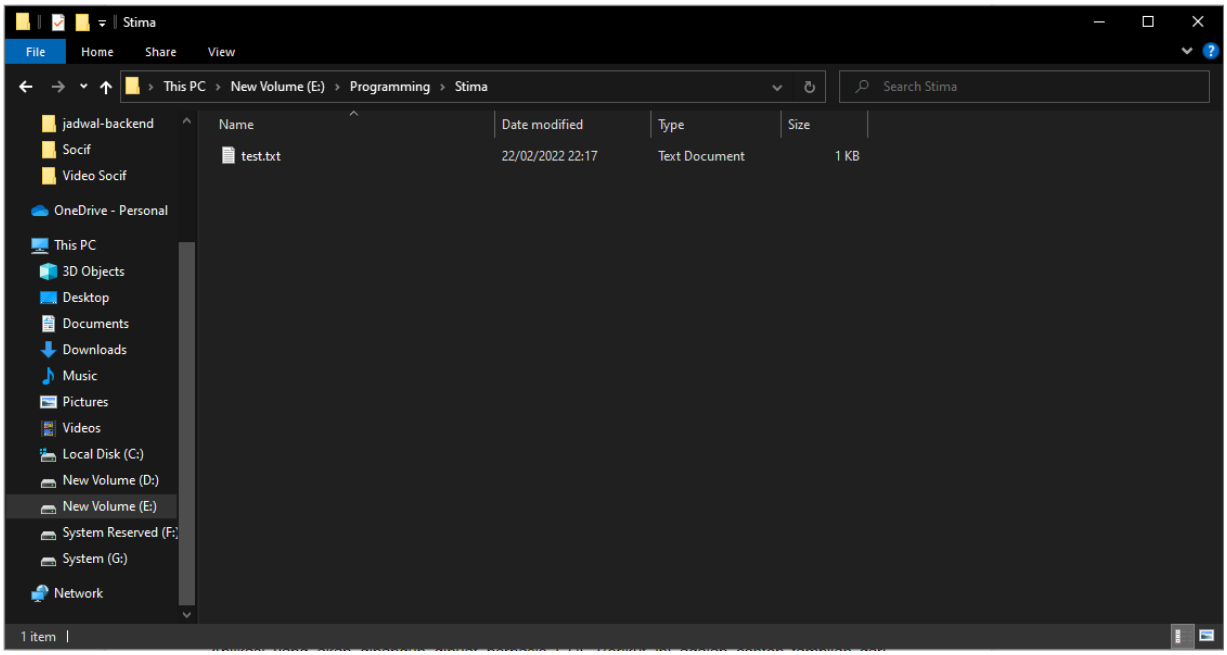
Gambar 4. Contoh output program jika file tidak ditemukan

Jika file yang ingin dicari pengguna tidak ada pada direktori file, misalnya saat pengguna mencari Kuis3Probstat.pdf, maka path pencarian DFS adalah sebagai berikut: C:\ → Folder1 → Folder3 → File1 → Folder3 → File2 → Folder3 → Folder1 → Folder4 → File3 → Folder4 → File4 → Folder4 → Folder1 → Folder5 → File5 → Folder5 → Folder6 → Kuis3_StrategiAlgoritma_13520000.pdf → Folder6 → Folder5 → Folder1 → C:\ → Folder2 → File6.
Pada gambar di atas, semua simpul dan cabang berwarna merah yang menandakan seluruh direktori sudah selesai diperiksa semua namun tidak ada yang mengarah ke tempat file berada.

Contoh Hyperlink Pada Path:



Contoh Hyperlink Dibuka Melalui Browser

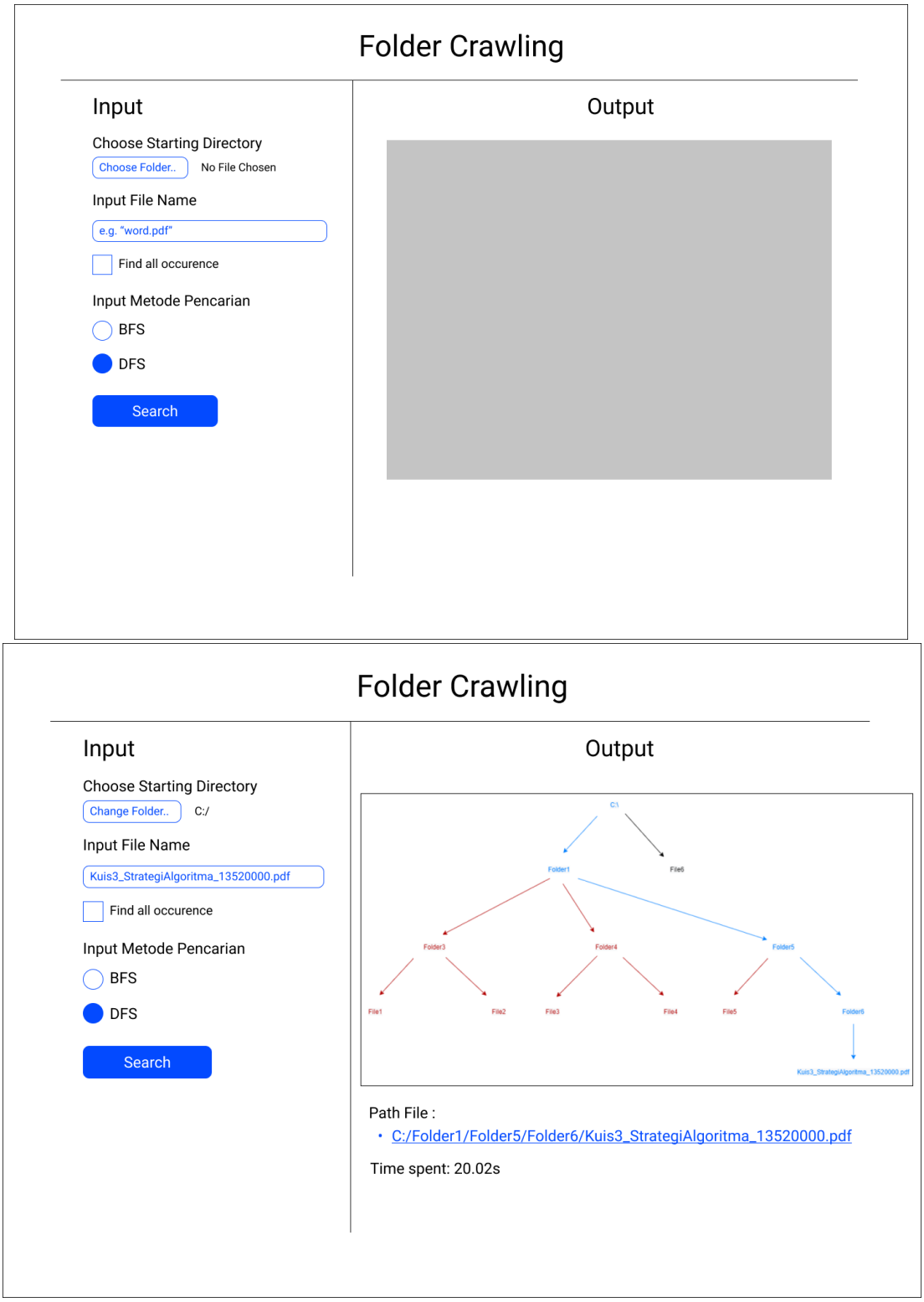


Contoh Hyperlink Dibuka Melalui Browser

Gambar 4. Contoh ketika hyperlink di-klik

Spesifikasi Program:

Aplikasi yang akan dibangun dibuat berbasis GUI. Berikut ini adalah contoh tampilan dari aplikasi GUI yang akan dibangun.



Gambar 9. Tampilan layout dari aplikasi desktop yang dibangun

Catatan: Tampilan diatas hanya berupa salah satu contoh layout dari aplikasi saja, untuk design layout aplikasi dibebaskan dengan syarat mengandung seluruh input dan output yang terdapat pada spesifikasi.

Spesifikasi GUI:

1. Program dapat menerima input folder dan query nama file.
2. Program dapat memilih untuk menampilkan satu hasil saja atau menemukan semua file yang memiliki nama file sama persis dengan input query
3. Program dapat memilih algoritma yang digunakan.
4. Program dapat menampilkan pohon hasil pencarian file tersebut dengan memberikan keterangan folder/file yang sudah diperiksa, folder/file yang sudah masuk antrian tapi belum diperiksa, dan rute folder serta file yang merupakan rute hasil pertemuan.
5. **(Bonus)** Program dapat menampilkan progress pembentukan pohon dengan menambahkan node/simpul sesuai dengan pemeriksaan folder/file yang sedang berlangsung.
6. Program dapat menampilkan hasil pencarian berupa rute/path (bisa lebih dari satu jika memilih menemukan semua file) serta durasi waktu algoritma.
7. GUI dapat dibuat **sekreatif** mungkin asalkan memuat 5(6 jika mengerjakan bonus) spesifikasi di atas.

Program yang dibuat harus memenuhi **spesifikasi wajib** sebagai berikut:

1) Buatlah program dalam bahasa **C#** untuk melakukan penelusuran *Folder Crawling* sehingga diperoleh hasil pencarian file yang diinginkan. Penelusuran harus memanfaatkan algoritma **BFS dan DFS**.

2) Awalnya program menerima sebuah input folder pada direktori yang ada dan nama file yang akan dicari oleh program.

3) Terdapat dua pilihan pencarian, yaitu:
    a. Mencari 1 file saja
       Program akan memberhentikan pencarian ketika sudah menemukan file yang memiliki nama sama persis dengan input nama file.
    b. Mencari semua kemunculan file pada folder root
       Program akan berhenti ketika sudah memeriksa semua file yang terdapat pada folder root dan program akan menampilkan daftar semua rute file yang memiliki nama sama persis dengan input nama file

4) Program kemudian dapat menampilkan **visualisasi pohon pencarian file** berdasarkan informasi direktori dari folder yang di-input. Pohon hasil pencarian file ini memiliki root adalah folder yang di-input dan setiap daunnya adalah file yang ada di folder root tersebut. Setiap folder/file direpresentasikan sebagai sebuah node atau simpul pada pohon. Cabang pada pohon menggambarkan folder/file yang terdapat di folder *parent*-nya.

   Visualisasi pohon juga harus disertai dengan **keterangan** node yang sudah diperiksa,

node yang sudah masuk antrian tapi belum diperiksa, dan node yang bagian dari rute hasil penemuan.

Proses visualisasi ini boleh memanfaatkan pustaka atau kakas yang tersedia. Sebagai referensi, salah satu kakas yang tersedia untuk melakukan visualisasi adalah **MSAGL** (https://github.com/microsoft/automatic-graph-layout) Berikut ini adalah panduan singkat terkait penggunaan MSAGL oleh tim asisten yang dapat diakses pada: https://docs.google.com/document/d/1XhFSpHU028Gaf7YxkmdbluLkQgVl3MY6gt1t-PL30LA/edit?usp=sharing

5) Program juga dapat menyediakan *hyperlink* pada setiap hasil rute yang ditemukan. *Hyperlink* ini akan membuka folder parent dari file yang ditemukan. Folder hasil *hyperlink* dapat dibuka dengan *browser* atau *file explorer*.

6) Mahasiswa **tidak diperkenankan** untuk melihat atau menyalin library lain yang mungkin tersedia bebas terkait dengan pemanfaatan BFS dan DFS. Tapi untuk algoritma lainnya seperti *string matching* dan akses *directory*, diperbolehkan menggunakan library jika ada.

Lain-lain:

1. Anda dapat menambahkan fitur-fitur lain yang menunjang program yang anda buat (unsur kreativitas).
2. Tugas dikerjakan berkelompok, minimal 2 orang dan maksimal 3 orang, boleh lintas kelas namun **tidak diperbolehkan sekelompok** dengan **orang yang sama dengan tubes-tubes stima sebelumnya**.
3. Semua kelompok harap mengisi data kelompok mereka pada link https://bit.ly/KelompokStima12022
4. Program dibuat dengan bahasa C# dengan kakas Visual Studio .NET, pelajarilah C# desktop development, **disarankan untuk menggunakan Visual Studio** untuk mempermudah pengerjaan.
5. Program harus modular dan mengandung komentar yang jelas.
6. Beri nama aplikasi anda tersebut dengan nama-nama yang menarik dan mudah diingat.
7. Dilarang menggunakan kode program yang diunduh dari Internet. Mahasiswa harus membuat program sendiri, tetapi belajar dari program yang sudah ada tidak dilarang.
8. Batas akhir pengumpulan tugas adalah **25 Maret 2022 23:59 WIB**. Keterlambatan dalam mengumpulkan akan diberi penalti pengurangan skor yang cukup signifikan.
9. Semua pertanyaan menyangkut tugas ini dapat dikomunikasikan lewat QnA yang bisa diakses pada https://bit.ly/QnAStima2022
10. **Bonus** (nilai maksimal 5): Setiap kelompok membuat video aplikasi yang mereka buat kemudian mengunggahnya ke Youtube. Video yang dibuat harus memiliki audio dan menampilkan wajah dari setiap anggota kelompok. Pada waktu demo aplikasi di depan asisten, mahasiswa mengakses video Youtube tersebut dan memutarnya di depan asisten. Beberapa contoh video tubes tahun-tahun sebelumnya dapat dilihat di YouTube dengan menggunakan kata kunci “Tubes Stima”, “Tugas besar stima”, “strategi algoritma”, dll.
11. Demo akan dilakukan, tunggu informasi lanjut setelah waktu pengerjaan tugas berakhir.

12. Setiap anggota kelompok harus memahami seluruh program, termasuk bagian yang bukan bagian mereka.
13. Program disimpan dalam folder **Tubes2_NIM** dengan NIM merupakan NIM anggota terkecil. Berikut merupakan struktur dari isi folder tersebut.
    a. Folder **src** berisi **source code**
    b. Folder **bin** berisi **executable** code / hasil build dari program C#
    c. Folder **doc** berisi **laporan tugas besar** dengan format **nama_kelompok.pdf**
    d. **README** untuk tata cara penggunaan yang minimal berisi:
        i. Deskripsi singkat program yang dibuat
        ii. Requirement program dan instalasi *module/package* tertentu bila ada
        iii. Langkah meng-*compile* program jika diperlukan
        iv. Cara menggunakan program
        v. Author / identitas pembuat
    e. Catatan untuk **README agar dibuat selengkap-lengkapnya**. Anggap asisten sebagai orang awam yang tidak tahu apa-apa. Jangan sampai asisten mencari tahu sendiri bagaimana cara menjalankan program Anda.
14. Program disimpan pada *repository* Github yang di-*private* sebelum deadline dan di-*public* setelah deadline pengumpulan. Form pengumpulan akan diberitahukan lebih lanjut oleh asisten.

Isi laporan:

- **Cover**: Cover laporan ada foto anggota kelompok (foto bertiga). Foto ini menggantikan logo “gajah” ganesha.
- **Bab 1**: Deskripsi tugas (dapat menyalin spesifikasi tugas ini).
- **Bab 2**: Landasan Teori.
    - Dasar teori (graph traversal, BFS, DFS) secara umum
    - Penjelasan singkat mengenai C# desktop application development
- **Bab 3**: Analisis Pemecahan Masalah.
    - Langkah-langkah pemecahan masalah
    - Proses mapping persoalan menjadi elemen-elemen algoritma BFS dan DFS.
    - Contoh ilustrasi kasus lain yang berbeda dengan contoh pada spesifikasi tugas
- **Bab 4**: Implementasi dan pengujian.
    - Implementasi program (*pseudocode* program utama).
    - Penjelasan struktur data yang digunakan dalam program dan spesifikasi program
    - Penjelasan tata cara penggunaan program (interface program, fitur-fitur yang disediakan program, dan sebagainya)
    - Hasil pengujian (*screenshot* antarmuka program dan beberapa data uji beserta skenario pengujian)
    - Analisis dari desain solusi algoritma BFS dan DFS yang diimplementasikan pada setiap pengujian yang dilakukan. Misalnya adalah apakah strategi DFS lebih baik dari BFS pada kasus kasus tertentu, dan analisis kalian mengenai mengapa hal itu bisa terjadi.
- **Bab 5**: Kesimpulan dan saran.
- Daftar Pustaka.

Keterangan laporan:

1. Laporan ditulis dalam bahasa Indonesia yang baik dan benar.
2. Laporan mengikuti format pada *section* “Isi laporan” dengan baik dan benar.
3. Identitas per halaman harus jelas (misalnya : halaman, kode kuliah).

Penilaian:

1. **Bagian 1** : Laporan (30%)
    a. Mapping persoalan ke dalam elemen algoritma BFS dan DFS (5%)
    b. Hasil pengujian dan analisis algoritma (10%)
    c. Komponen-komponen lain dalam laporan (10%)
    d. Kesesuaian laporan dengan program (5%)
2. **Bagian 2** : Implementasi Program (70%)
    a. Kebenaran program (30%)
    b. Pemahaman terhadap cara kerja program (25%)
    c. Kreativitas pembuatan GUI (15%)
3. **Bagian 3** : Bonus (10%)
    a. Mengimplementasikan tampilan *progres* pembentukan pohon (5%)
    b. Membuat video demonstrasi program (5%)

--- *Selamat Mengerjakan!* —

“Inget, cari file bukan cari jodoh :D”
– Tito –

“Boleh dicoba tuh cari jodoh pake BFS DFS”
– Hokki –

“Jangan lelah dulu ya gais, baru setengah jalan :’)”
– Jojo –

“Semoga ketemu apa yang dicari”
- Girvin-

“Hati-hati salah jalan, jangan sampai nyasar”
– Christo –

“Kalau lelah jangan lupa ~~breadth~~ breathe first gais”
– Dzaki –

“Ya ndak tau, kok tanya saya”
– Dimas –

“You don’t have to be at the same node as someone else.
You can just be at your node and that is okay too👌”
– Shafira –