|  |
| --- |
| **LABORATORIUM: TELEMATIKA**  **POLITEKNIK NEGERI LHOKSEUMAWE**  **PENGUJIAN : *BUBBLE SORT ALGORITHM*** |

**I. Capaian Praktikum/Kompetensi**

* Mahasiswa dapat menjelaskan cara kerja algoritma *bubble sort.*
* Mahasiswa dapat mengimplementasikan algoritma *bubble sort* dalam bentuk program.
* Mahasiswa dapat menerapkan algoritma *bubble sort* dalam kasus sederhana.

**II. Keselamatan Kerja**

* Praktikum hanya dapat dilakukan atas petunjuk dosen pengasuh atau instruktur.
* Pastikan semua peralatan dalam keadaan baik sebelum dan sesudah melakukan praktikum serta menjaga semua peralatan yang digunakan.
* Perlu mematuhi semua peraturan pada laboratorium yang digunakan
* Sebelum melakukan praktikum agar selalu mengawali dengan berdoa

**III. Teori**

Pengurutan (*sorting*) adalah proses mengatur sekumpulan obyek menurut urutan atau susunan tertentu. Urutan tersebut dapat menaik (*ascending*) atau menurun (*descending*). Jika diberikan n buah elemen disimpan di dalam larik L, maka :

- pengurutan menaik adalah L[0] < L[1] < L[2] < … < L[n-1]

- pengurutan menurun adalah L[0] > L[1] > L[2] > … > L[n-1]

Bermacam-macam metode yang dipakai untuk melakukan pengurutan, antara lain *Bubble Sort*, *Selection Sort*, *Insertion Sort*, *Shell Sort*, *Quick Sort*, *Merge Sort.* Pada jobsheet ini khusus dibahas mengenai *bubble sort.*

Algoritma *bubble sort* ini diinspirasi oleh gelembung sabun yang berada di permukaan air. Karena berat jenis gelembung sabun lebih ringan dibandingkan dengan berat jenis air, sehingga gelembung sabun selalu terapung di permukaan air. Prinsip pengapungan inilah yang diterapkan ke Algoritma ini, dimana nilai yang paling rendah berada di posisi paling atas, melalui proses pertukaran.

Konsep dasar dari algoritma ini adalah setiap data yang ada di kumpulan, dibandingkan dengan data-data lainnya, artinya jika jumlah data sebanyak 5, maka akan terjadi perbandingan sebanyak (5-1)2 = 16 kali. Atau secara umum dapat ditarik rumus, untuk jumlah data sebanyak n buah, maka :

Jumlah iterasi pembandingan = (n-1)2 (1)

Jika data-data tersebut disimpan di dalam larik L, maka :

1. Untuk pengurutan menaik, pembandingnya sebagai berikut : L[n] < L{n-1]

2. Untuk pengurutan menurun, pembandingnya sebagai berikut : L[n] > L[n-1]

Jika kondisi diatas terpenuhi, maka data yang ada di indeks n-1 akan ditukar dengan data yang ada di indeks n. Contoh: Misalkan terdapat sebuah larik/array yang memiliki 4 buah elemen seperti tampak pada Gambar 1. Untuk memahami proses pengurutan pada algoritma *bubble sort*, Perhatikan langkah-langkah dibawah ini.

Indeks 🡪 0 1 2 3

|  |  |  |  |
| --- | --- | --- | --- |
| 70 | 40 | 50 | 10 |

Elemen 🡪 1 2 3 4

Gambar 1. Elemen Array sebelum *sorting*

Langkah-langkah pengurutan menaik, menggunakan algoritma *buble sort.*

**Siklus ke-1 (*Pass-1*).**

input pada siklus ke-1 ( pass-1) adalah :

|  |  |  |  |
| --- | --- | --- | --- |
| 70 | 40 | 50 | 10 |

Gambar 2. Elemen Array pada awal Pass-1

1. Bandingkan elemen 1 dengan elemen 2, jika elemen ke-1 lebih besar dari elemen ke-2 (L[0] >L[1]), maka akan membentuk operasi *swap*, yang berarti terjadi pertukaran tempat (*interchange*) antara elemen ke-2 dan elemen ke-1. Jika tidak maka, tidak terjadi pergantian posisi.

L[0] > L[1]) ? 🡪 70 > 40 ?

jawabannya benar, berarti terjadi operasi swap, elemen ke 2 dan elemen ke 1 berpindah tempat, sehingga urutan data pada larik menjadi:

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 70 | 50 | 10 |

Gambar 3. Elemen Array pada awal Pass-1, langkah ke-1

1. Berikutnya elemen ke 2 dibandingkan dengan elemen ke 3.

L[1] > L[2]) ? 🡪 70 > 50 ?

jawabannya benar, berarti terjadi operasi swap, elemen ke 3 dan elemen ke 2 berpindah tempat, sehingga urutan data pada larik menjadi:

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 50 | 70 | 10 |

Gambar 4. Elemen Array pada awal Pass-1, langkah ke-2

1. Berikutnya elemen ke 3 dibandingkan dengan elemen ke 4.

L[2] > L[3]) ? 🡪 70 > 10 ?

jawabannya benar, berarti terjadi operasi swap, elemen ke 4 dan elemen ke 3 berpindah tempat, sehingga urutan data pada larik menjadi:

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 50 | 10 | 70 |

Gambar 5. Elemen Array pada akhir Pass-1.

Pada akhir pass-1, nilai terbesar dalam hal ini 70 sudah berada di posisi yang seharusnya (berada di paling akhir).

**Siklus ke-2 ( *Pass-2*)**

input pada siklus ke-2 ( pass-2) adalah :

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 50 | 10 | 70 |

Gambar 6. Elemen Array pada awal Pass-2.

1. Bandingkan elemen ke 1 dengan elemen ke 2 (data pada indeks 0 dengan data pada indeks 1).

L[0] > L[1]) ? 🡪 40 > 50 ?

jawabannya salah, berarti tidak terjadi operasi swap, tidak terjadi perubahan posisi (*no change*) dari kedua nilai yang dibandingkan, sehingga urutan data pada larik menjadi tetap:

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 50 | 10 | 70 |

Gambar 7. Elemen Array pada awal Pass-2, langkah ke-1

1. Berikutnya elemen ke 2 dibandingkan dengan elemen ke 3.

L[1] > L[2]) ? 🡪 50 > 10 ?

jawabannya benar, berarti terjadi operasi swap, elemen ke 3 dan elemen ke 2 berpindah posisi, sehingga urutan data pada larik menjadi:

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 10 | 50 | 70 |

Gambar 8. Elemen Array pada awal Pass-2, langkah ke-2

1. Berikutnya elemen ke 3 dibandingkan dengan elemen ke 4.

L[2] > L[3]) ? 🡪 50 > 70 ?

jawabannya salah, berarti tidak terjadi operasi swap, tidak terjadi perubahan posisi (*no change*) dari kedua nilai yang dibandingkan, sehingga urutan data pada larik menjadi tetap:

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 10 | 50 | 70 |

Gambar 9. Elemen Array pada akhir Pass-2.

Pada akhir pass-2, nilai terbesar ke dua dalam hal ini 50 sudah berada di posisi yang seharusnya .

**Siklus ke-3 (*pass-3*)**

input pada siklus ke-3 ( pass-3) adalah :

|  |  |  |  |
| --- | --- | --- | --- |
| 40 | 10 | 50 | 70 |

Gambar 10. Elemen Array pada awal Pass-3.

1. Bandingkan elemen ke 1 dengan elemen ke 2 (data pada indeks 0 dengan data pada indeks 1).

L[0] > L[1]) ? 🡪 40 > 10 ?

jawabannya benar, berarti terjadi operasi swap, elemen ke 2 dan elemen ke 1 berpindah tempat atau posisi, sehingga urutan data pada larik menjadi:

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | 40 | 50 | 70 |

Gambar 11. Elemen Array pada awal Pass-3, langkah ke-1

1. Berikutnya elemen ke 2 dibandingkan dengan elemen ke 3.

L[1] > L[2]) ? 🡪 40 > 50 ?

jawabannya salah, berarti tidak terjadi operasi swap, tidak terjadi perubahan posisi (*no change*) dari kedua nilai yang dibandingkan, sehingga urutan data pada larik menjadi tetap:

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | 40 | 50 | 70 |

Gambar 12. Elemen Array pada awal Pass-3, langkah ke-2

1. Berikutnya elemen ke 3 dibandingkan dengan elemen ke 4.

L[2] > L[3]) ? 🡪 50 > 70 ?

jawabannya salah, berarti tidak terjadi operasi swap, tidak terjadi perubahan posisi (*no change*) dari kedua nilai yang dibandingkan, sehingga urutan data pada larik menjadi tetap:

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | 40 | 50 | 70 |

Gambar 13. Elemen Array pada akhir Pass-3.

Pada akhir pass-3, nilai pada larik/array seluruhn ya sudah terurut.

Dari uraian diatas, dapat dilihat bahwa pada larik/array yang memiliki 4 elemen, maka terdapat 3 pass (pass-1, pass-2, dan pass-3), berarti banyaknya pass pada larik/array yang memiliki n elemen adalah sebanyak (n – 1).

**IV. Alat/Bahan**

1. Komputer Pribadi (*Personal Computer*)
2. Perangkat Lunak Dev-C++, gcc, vim atau pico
3. Sistem operasi *windows* atau *linux*

**V. Prosedur Praktikum**

1. Buatlah flowchart untuk mengilustrasikan proses pengurutan pada *bubble sort* berdasarkan langkah langkah yang telah dijelaskan diatas.
2. Buatlah program *bubble sort* berdasarkan langkah langkah yang telah dijelaskan diatas, untuk pengurutan menaik (*ascending*).
3. Buatlah program *bubble sort* berdasarkan langkah langkah yang telah dijelaskan diatas, untuk pengurutan menurun (*descending*).

**VI. Data Percobaan**

Buat program seperti tugas diatas dan hasil eksekusinya dalam borang berikut:

Tabel 1. Program dan Hasil Eksekusi

|  |  |  |
| --- | --- | --- |
| **No Perc.** | **Program (*Source Code*)** | **Hasil (*Output*)** |
| 1.  2.  3. |  |  |

**VII. Analisa dan Kesimpulan**

1. Analisa hasil dari eksekusi program diatas, berikan maksud dari setiap baris program yang diberikan.
2. Dari hasil percobaan yang dilakukan, buat kesimpulan dari masing-masing percobaan diatas.

**VIII. Daftar Pustaka**

Jain, H. (2017). Problem Solving in Data Structures & Algorithms Using C. Bhopal, India.

Tutorialspoint. (2016). Data Structures & Algorithms Simply Easy Learning, Tutorial points Ltd, www.tutorialspoint.com.