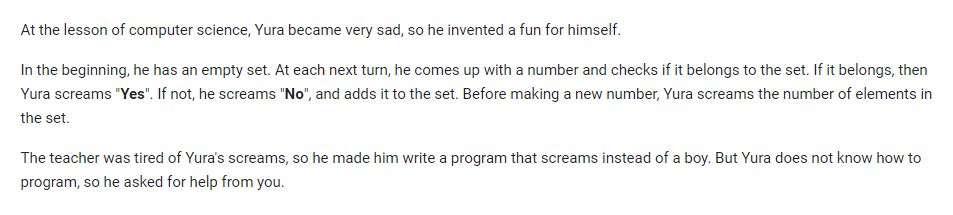
Tugas 5 PBO C - 1

Brendan Timothy Mannuel

5025221177

Interesting Fun of Yura



Untuk soal ini kita diperlukan untuk mencari apakah suatu bilangan telah tersimpan didalam suatu set yang berisi banyak angka, dimana angka akan diberikan oleh soal. Kita harus mengeluarkan “Yes” jika bilangan sudah terdapat di dalam set, dan “No” jika bilangan belum terdapat didalam set dan kita akan memasukannya, setelah itu kita juga harus mengeluarkan banyak bilangan yang tersimpan didalam set. Pada soal ini kita dapat menggunakan implementasi dari class SET di c++.

A screenshot of a computer

Description automatically generated

Set termasuk sebagai non linear class di stl pada c++, dimana non linear itu berarti bahwa kita tidak harus melakukan pengecekan secara satu satu untuk menyampai target, salah satu contoh utama dari non linear adalah graph dimana dalam 1 node dapat berhubungan dengan banyak node yang lain. Kenapa kita menggunakan set karena kita akan memanfaatkan fungsi dari set yaitu count() dan size()A screenshot of a computer

Description automatically generated

count() yang akan mencari suatu value yang sesuai dan akan mereturn angka 1 ketika ketemu dan angka 0 jika tidak menemukan.

A blue and white rectangular object

Description automatically generated

serta size() yang akan mengembalikan besar ukuran set tersebut  
A computer screen shot of a program code

Description automatically generated

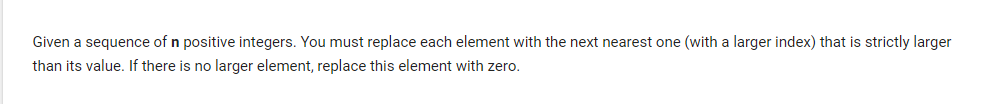
Kode akan mengecek apakah angka yang diinput ada dalam set, jika tidak ada maka kita akan mengeluarkan “No” dan memasukan angka tersebut kedalam set, serta mengeluarkan ukuran set. Jika terdapat maka kita akan mengeluarkan “Yes” dan besar dari set tanpa harus menginput angkanya.

Tugas 5 PBO C - 2

Brendan Timothy Mannuel

5025221177

Replacement



Pada soal ini kita akan mengganti suatu angka dengan angka selanjutnya yang lebih besar tetapi harus berada di kanan nya. Kita akan melakukan 2 pendekatan, yaitu secara linear dan non linear.

Untuk cara linear kita akan menggunakan class stack

A screenshot of a computer program

Description automatically generated

Stack adalah sebuah class linear yang menyimpan suatu value secara LIFO atau last in first out.

A black screen with white text

Description automatically generated

Selain menggunakan stack, kita akan menggunakan bantuan class vector sebagai tempat penyimpanan awal dan hasil akhir, kita akan mengisi vector hasil dengan angka 0 terlebih dahulu

A black screen with white text

Description automatically generated

Kita akan menginputkan deretan angka kedalam vector sequence

A computer screen with white text

Description automatically generated

Loop ini adalah bagian penting dari algoritma yang menghitung elemen-elemen berikutnya yang lebih besar dari setiap elemen dalam urutan masukan dan menyimpan hasilnya dalam vektor res.

1. for (int i = 0; i < n; i++): Loop ini akan berjalan sebanyak n kali, di mana n adalah panjang urutan yang diinputkan.
2. while (!s.empty() && sequence[i] > sequence[s.top()]): Loop ini adalah bagian dari algoritma yang mencari elemen-elemen berikutnya yang lebih besar. Ini akan terus berjalan selama stack s tidak kosong dan elemen saat ini dalam urutan sequence[i] lebih besar daripada elemen yang terdapat di puncak stack.
3. res[s.top()] = sequence[i];: Ketika loop di atas terus berjalan, itu berarti elemen di puncak stack saat ini memiliki elemen berikutnya yang lebih besar. Oleh karena itu, nilai dari elemen ini dalam vektor res diisi dengan nilai sequence[i].
4. s.pop();: Setelah nilai res telah diisi, elemen di puncak stack saat ini dihapus dari stack.
5. s.push(i);: Elemen saat ini (indeks i) ditambahkan ke stack untuk mencari elemen yang lebih besar selanjutnya untuk elemen ini.

Dengan cara ini, algoritma ini akan mengisi vektor res dengan nilai-nilai yang merupakan elemen-elemen berikutnya yang lebih besar dari setiap elemen dalam urutan sequence.

A black background with colorful letters

Description automatically generated

Setelah itu kita akan mengoutputkan hasilnya

A screenshot of a computer

Description automatically generated

Untuk pendekatan Kedua kita akan menggunakan pendekatan secara non linear dengan menggunakan set.

A screenshot of a computer

Description automatically generated

Untuk yang pertama kita akan menggunakan 2 set, yang pertama sebagai iterator dan yang kedua sebagai tempat penyimpanan sementara, serta kita akan menggunakan 2 array

A screen shot of a computer code

Description automatically generated

Kemudian kita akan masuk kedalam loop utama

A computer screen with white text and numbers

Description automatically generated

Kita akan menginput deretan angka kedalam array sequence, kemudain kita akan memulai array dari kanan menuju kiri. Kemudaian kita akan memasukan angka sekarang kedalam set, kemudian kita akan membuat iterator menunjuk kepada angka sekarang dan kita akan menghapus semua angka yang lebih kecil dari sekarang. Kemudian kita akan memasukan elemen kedua (angka yang lebih besar selanjutnya) kedalam array result, jika tidak ada maka kita akan memasukan angka 0. Kemudian kita akan mengoutput array result jika loop telah selesai.

A screen shot of a computer

Description automatically generated

KESIMPULAN

Dari kedua cara yang dipakai terlihat bahwa cara linear menggunakan stack lebih cepat dibandingkan dengan cara non linear menggunakan set. Hal ini disebabkan oleh berapa hal, salah satunya adalah tipe soal, dimana soal ini memrlukan adanya pengecekan secara satu persatu dimana tentunya cara linear pasti akan jauh lebih cepat karena cara native dari linear adalah melakukan checking satu satu, dimana non linear tidak melakukan checking satu satu menyebabkan pasti adanya step tambahan yang menyebabkan lebih banyak waktu yang terbuang. Hal ini membuktikan bahwa class yang dipakai juga harus mengikuti permasalahan dari soal, dibawah ini ada beberapa perbedaan antara linear dan non linear:

|  |  |
| --- | --- |
| Linear | Non-Linear |
| Dalam struktur ini, elemen-elemen disusun secara berurutan atau linear dan melekat satu sama lain. | Dalam struktur ini, elemen-elemen disusun secara hirarkis atau non-linear. |
| Array, linked list, stack, dan queue | Trees dan graphs |
| Karena linear, mereka mudah untuk diimplementasikan | Karena organisasi non-linear, mereka sulit untuk diimplementasikan. |
| Karena struktur data linear memiliki 1 level, maka dibutuhkan satu kali jalan untuk menelusuri setiap item data. | Item data dalam struktur data non-linear tidak dapat diakses dalam sekali traverse. Diperlukan beberapa traversing untuk melintasinya. |
| Setiap item data terhubung dengan item sebelumnya dan berikutnya | Setiap item terhubung dengan banyak item lainnya. |
| Struktur data ini tidak mengandung hirarki apa pun, dan semua elemen data diorganisasi dalam satu tingkat. | Dalam hal ini, elemen-elemen data disusun dalam beberapa tingkat. |
| Kompleksitas waktu dari struktur data linear meningkat mengikuti ukuran input. | Kompleksitas waktu dari struktur data non-linear seringkali tetap sama dengan peningkatan ukuran input. |
| Dalam hal ini, penggunaan memori tidak efisien. | Dalam hal ini, memori digunakan dengan cara yang sangat efisien. |