Guilherme Montalbano Volante, RA 75155, EC 3° semestre

Estrutura de dados I

Relatório espaço diversificado

Códigos e Prints

https://github.com/Gmontalbano/espacodiversidicado

Vídeos 1 – 6: criação do ambiente e entendendo o básico do C++

O curso começa nos ensinando a arrumar o compilador em nossas máquinas, basta saber seu sistema operacional e configurar de acordo com as indicações do curso.

Começa a nos ensinar a como fazer uma saída para o usuário, nos explica os tipos de dados INT, FLOAT, DOUBLE e etc. Mostra as operações básicas, como soma, multiplicação e divisão.

Vídeos 21 – 27: ponteiros e constantes

Começa o curso explicando sobre o ponteiro, falando que o ponteiro é uma variável que contém o endereço de outra variável na memória do computador. É mostrado alguns exemplos de como acessar o valor da variável ou o endereço, utilizando “*” e “&”.

O primeiro código é simples, apenas criamos uma variável e pegamos seu valor através de um ponteiro, depois utilizamos uma função void para alterar o valor da variável, que só é possível se utilizarmos ponteiros pois uma função void não retorna valores. Quando chamamos uma função void para alterar o valor nós não podemos passar apenas uma variável comum, pois assim apenas passamos seu valor e isso será atribuído a variável dentro do escopo da função e não no escopo do main, por isso podemos passar um ponteiro, pois assim passamos um endereço de memória que deverá ter seu valor armazenado alterado.

Nos explica como utilizar o for em um vetor e utilizar o sizeof, pois o tamanho pode não ser padrão entre os vetores.

Apresenta a função delete a qual eu não sabia da existência e achei muito útil, para podermos apagar algo da memória que não utilizamos mais, e assim não ficar ocupando espaço atoa podendo deixar o processo mais lento.

Utilizando um while i<tamanho, podemos navegar tranquilamente pelo array, sem a necessidade de colocarmos a posição e apenas utilizando ponteiro++, pois como o ponteiro recebe um endereço de memória quando é passado ponteiro++ ele aumenta um no endereço de memória e não no valor que a variável guarda.

Vídeos 28 – 30: structs e classes

Podemos armazenar dados de forma organizada com struct e classes, por que utilizamos? Simples, imagina um sistema básico de uma escola, cada aluno com seu nome, idade, RA e sala. se nós fossemos organizar em vetores teríamos 4 vetores distintos e cada um guardando uma informação de cada aluno, em um sistema com 5 alunos ok é possível fazer assim, mas imagina que você tenha 4000 alunos, como você vai saber qual a idade de tal aluno se não existe nada ligando o vetor nome com o vetor idade. Para isso temos as estruturas, que conseguimos atribuir para a estrutura aluno seus elementos, nome, idade, RA e sala, assim quando pegamos um aluno x temos todos seus dados de forma organizada.

temos alguns dados sensíveis quando trabalhamos com classes, como por exemplo em um sistema de banco, saldo, senhas, CPF, por isso podemos definir se os atributos das classes serão públicos - Qualquer um pode acessar ou privados - só pode ser acessado com o acesso certo, deixando assim o sistema mais seguro.

Vídeos 63 – 67: listas encadeadas

Listas encadeadas é uma estrutura com valores com ponteiros que ficam ligados entre si, primeiro é apontado o ínicio e o fim da fila, e cada estrutura vai apontar para o endereço do próximo elemento, o último elemento irá apontar para NULL significando que ali é o final da lista.

Quando inserimos um elemento novo em uma lista precisamos observar algumas regrinhas, a lista possui algum elemento? Caso não, é preciso indicar que aquele elemento é o início e também o final da lista, caso possua o valor precisamos observar onde nós vamos inserir, no começo? precisamos trocar então

para onde o inicio aponta, no final, precisamos pegar o final que apontava para NULL e apontar para o novo elemento e o novo elemento começa a apontar para NULL, no meio, pega-se o anterior da posição desejada e o novo elemento aponta para onde ele apontava e agora o anterior passa a apontar para o novo elemento assim “abrimos espaço” no encadeação da lista, se sem querer passarmos uma posição que não exista? precisamos retornar que aquela posição não existe, pois será impossível criar os elos até a posição, pois eles não irão existir.