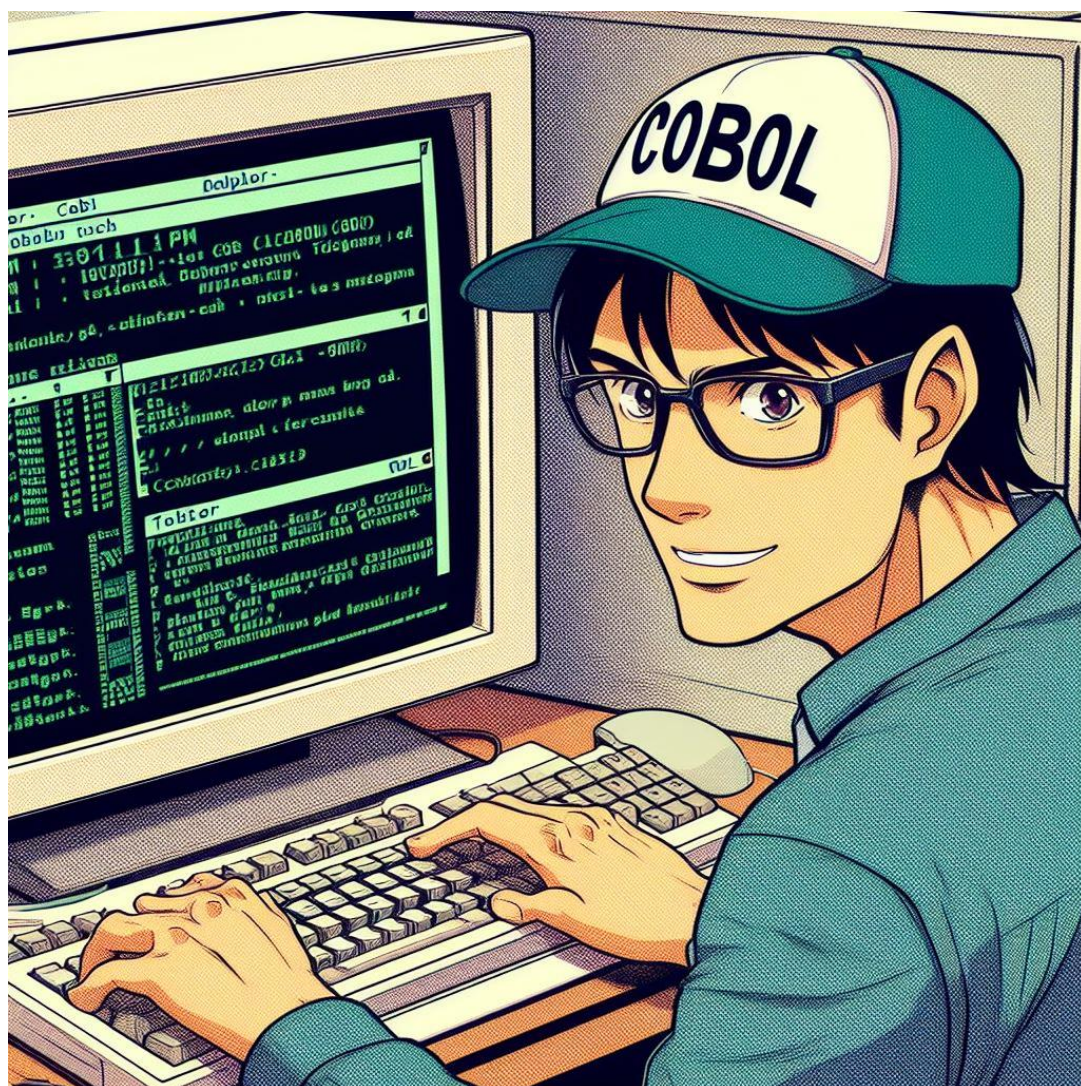
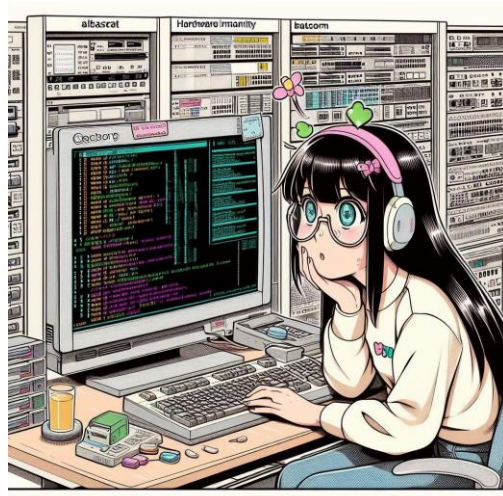


Intimidade com o Hardware

Alocando Memória

Dinamicamente





Intimidade com o Hardware

Alocando Memória Dinamicamente

Introdução

COBOL é uma linguagem de programação que tem sido amplamente utilizada em mainframes para desenvolvimento de sistemas de negócios e processamento de dados. Neste guia, vamos explorar alguns conceitos fundamentais de COBOL, com foco na alocação dinâmica de memória e estruturas de dados como listas e árvores.

1. Alocação Dinâmica de Memória

A alocação dinâmica de memória em COBOL permite que o programa aloque e libere memória em tempo de execução. Isso é útil quando não se sabe o tamanho exato dos dados necessários antecipadamente. Um exemplo de código em COBOL para alocação dinâmica de memória é:



```
IDENTIFICATION DIVISION.
```

```
PROGRAM-ID. AlocaMemoria.
```

```
DATA DIVISION.
```

```
WORKING-STORAGE SECTION.
```

```
01 PtrMemoria POINTER.
```

```
PROCEDURE DIVISION.
```

```
MOVE 100 TO PtrMemoria.
```

```
DISPLAY "Memória alocada com sucesso!".
```

```
STOP RUN.
```

2. Lista Circular

Uma lista circular é uma estrutura de dados na qual o último nó aponta de volta para o primeiro nó, formando um ciclo. Aqui está um exemplo de uma lista circular com alocação dinâmica de memória em COBOL:



```
IDENTIFICATION DIVISION.  
PROGRAM-ID. ListaCircular.  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 PtrInicio POINTER.  
PROCEDURE DIVISION.  
    PERFORM VARYING i FROM 1 BY 1 UNTIL i > 15  
        ALLOCATE 100 TO PtrNovo  
        SET END OF PtrNovo TO PtrInicio  
        SET PtrInicio TO PtrNovo  
    END-PERFORM.  
    DISPLAY "Lista circular criada com sucesso!".  
    STOP RUN.
```

3. Lista Circular Duplamente Encadeada


Uma lista circular duplamente encadeada é semelhante a uma lista circular, mas cada nó tem um ponteiro tanto para o próximo quanto para o nó anterior. Veja um exemplo em COBOL:



```
IDENTIFICATION DIVISION.  
PROGRAM-ID. ListaCircularDupla.  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 PtrInicio POINTER.  
PROCEDURE DIVISION.  
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 15  
        ALLOCATE 100 TO PtrNovo  
        SET END OF PtrNovo TO PtrInicio  
        SET PtrInicio TO PtrNovo  
    END-PERFORM.  
STOP RUN.
```


4. Árvore Binária

Uma árvore binária é uma estrutura de dados na qual cada nó tem no máximo dois filhos. Aqui está um exemplo de árvore binária com alocação dinâmica de memória em COBOL:



```
IDENTIFICATION DIVISION.  
PROGRAM-ID. ArvoreBinaria.  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 PtrRaiz POINTER.  
PROCEDURE DIVISION.  
    PERFORM VARYING i FROM 1 BY 1 UNTIL i > 15  
        ALLOCATE 100 TO PtrNovo  
        IF PtrRaiz = NULL  
            SET PtrRaiz TO PtrNovo  
        ELSE  
            PERFORM INSERIR-NO  
        END-IF  
    END-PERFORM.  
    DISPLAY "Árvore binária criada com sucesso!".  
    STOP RUN.  
INSERIR-NO.  
    * Lógica para inserir um nó na árvore binária.  
    EXIT.
```

5. Árvore Balanceada

Uma árvore balanceada é uma árvore binária na qual a altura das sub-árvores adjacentes nunca difere em mais de uma unidade. Aqui está um exemplo de uma árvore balanceada com alocação dinâmica de memória em COBOL:



```
IDENTIFICATION DIVISION.  
PROGRAM-ID. ArvoreBalanceada.  
  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 PtrRaiz POINTER.  
  
PROCEDURE DIVISION.  
    PERFORM VARYING i FROM 1 BY 1 UNTIL i > 15  
        * Lógica para inserir um nó na árvore balanceada.  
    END-PERFORM.  
  
    DISPLAY "Árvore balanceada criada com sucesso!".  
    STOP RUN.
```



Conclusão

Este guia abordou os conceitos de COBOL para Mainframe, com foco em alocação dinâmica de memória e estruturas de dados como listas e árvores. Esperamos que este material ajude os desenvolvedores júnior a entenderem melhor como trabalhar com COBOL em ambientes de mainframe.