UFAM

FT – FACULDADE DE TECNOLOGIA
ENGENHARIA DA COMPUTAÇÃO

JEAN CLEISON BRAGA GUIMARÃES – 21601227

TRABALHO 1: ARVORES AVL (Adelson-Velskii e Landis) – IMPLEMENTAÇÃO DE UM PROGRAMA DE MARCAÇÃO DE CONSULTAS

MANAUS, AM
2019

JEAN CLEISON BRAGA GUIMARÃES

TRABALHO 1: ARVORES AVL (Adelson-Velskii e Landis) – IMPLEMENTAÇÃO DE UM PROGRAMA DE MARCAÇÃO DE CONSULTAS

O trabalho foi solicitado pelo professor de Algoritmos e Estrutura de Dados II, Edson Nascimento para obtenção de nota parcial por parte dos alunos no primeiro semestre de 2019.

MANAUS, AM
2019

Introdução

Árvore AVL é uma árvore binária de busca balanceada, ou seja, uma árvore balanceada (árvore completa) são as árvores que minimizam o número de comparações efetuadas no pior caso para uma busca com chaves de probabilidades de ocorrências idênticas. Contudo, para garantir essa propriedade em aplicações dinâmicas, é preciso reconstruir a árvore para seu estado ideal a cada operação sobre seus nós (inclusão ou exclusão), para ser alcançado um custo de algoritmo com o tempo de pesquisa tendendo a *O(log n).*

As operações de busca, inserção e remoção de elementos possuem complexidade *O(log n)* (no qual *n* é o número de elementos da árvore), que são aplicados a árvore de busca binária.

O nome AVL vem de seus criadores soviéticos Adelson Velsky e Landis, e sua primeira referência encontra-se no documento "Algoritmos para organização da informação" de 1962.

Nessa estrutura de dados cada elemento é chamado de nó. Cada nó armazena uma chave e dois ponteiros, uma para a subárvore esquerda e outro para a subárvore direita.

No seguinte trabalho, foi usado uma árvore AVL para o armazenamento dos dados, mas decidi abstrair o conceito e funções relacionadas à árvore e foquei na construção de funções visando simplificar o conteúdo e facilitar a implementação do programa.

Implementação

- O programa foi construído para a marcação de consultas afim de facilitar a administração *semanal* dos 4 consultórios da clínica.

- Este pode apresentar relatórios mostrando as consultas agendadas no dia escolhido, fazer marcação de novas consultas e buscar informações a respeito de uma consulta já agendada.

- Há ainda uma função de testes, que gera aleatoriamente uma série de consultas.

Todo o programa é baseado no uso de códigos de consultas, que são gerados na hora da marcação, ou aleatoriamente pelo Gerador de Testes. Usei como inspiração o conceito de *Hashing.*

O código gerado é um número de 5 dígitos.
Sendo esta a composição:

1º Dígito - Dia da Consulta
(2 - Segunda-Feira, 3 - Terça-Feira, 4 - Quarta-Feira, 5 - Quinta-Feira, 6 - Sexta-Feira).

2º e 3º Dígitos - Horário da Consulta.

4º e 5º Dígitos - Número do Consultório.

Ex: 51403.

Para o código acima, o algoritmo entende que esta consulta está marcada para:

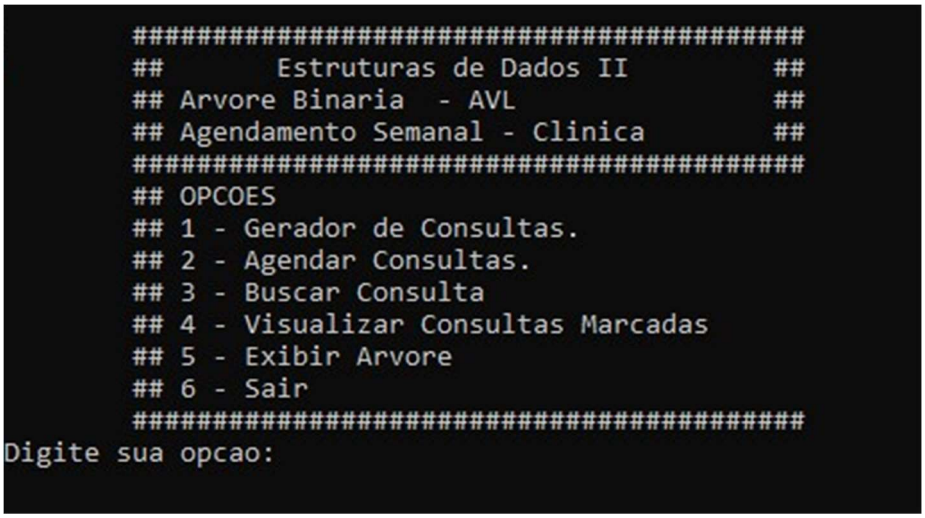
Quinta-Feira, às 14h no Consultório 03

Supondo que o usuário não saiba o significado, basta fazer uma busca no programa utilizando o código de sua consulta que será informado as informações da mesma.

Para facilitar, chamaremos esse código de **COD**.

Execução do Algoritmo

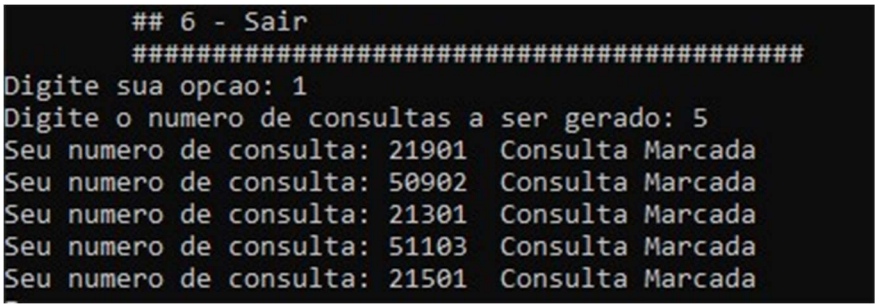
O programa, ao ser executado, está encarregado de imprimir na tela um menu, que por sua vez, irá mostrar as opções disponíveis ao usuário.



A seguir há a explicação de cada opção:

1 - Gerador de Consultas:
Quando o usuário informa o número de testes que deseja, o programa gera os **CODs** aleatoriamente e, se disponível, faz o agendamento.

Tela da execução:



2 - Agendar Consulta:

- Disponível para o usuário agendar as consultas, informando o dia e horário desejado. O programa busca um consultório disponível, e retorna ao usuário o código de consulta.
- Caso não houver, o programa informa que não há disponibilidade no horário desejado.

Tela de execução:

Se disponível:

```
Digite sua opcao: 2

2 - segunda-feira       3 - terca-feira         4 - quarta-feira        5 - quinta-feira        6 - sexta-feira

Escolha o dia da sua consulta: 3
Digite o horario que deseja ser atendido(9h-21h):20

Seu numero de consulta: 32001  Consulta Marcada
```

Se indisponível:

```
Digite sua opcao: 2

2 - segunda-feira       3 - terca-feira         4 - quarta-feira        5 - quinta-feira        6 - sexta-feira

Escolha o dia da sua consulta: 3
Digite o horario que deseja ser atendido(9h-21h):20

Horario Indisponivel. Tente em outro horario.
```

3 - Buscar Consulta:

- Busca as informações da consulta através do **COD**. Se não existir, informa ao usuário.

Tela de execução:

Se estiver marcada:

```
Digite sua opcao: 3
Digite o codigo da consulta: 32001

Consulta marcada para terca-feira
Horario: 20h
Consultorio: 1
```

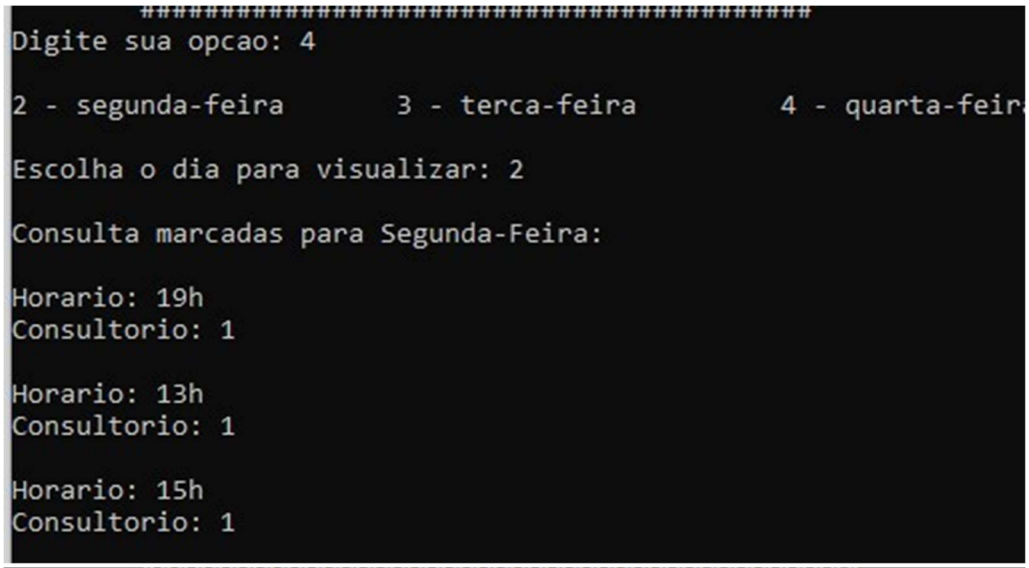
Se não estiver marcada:

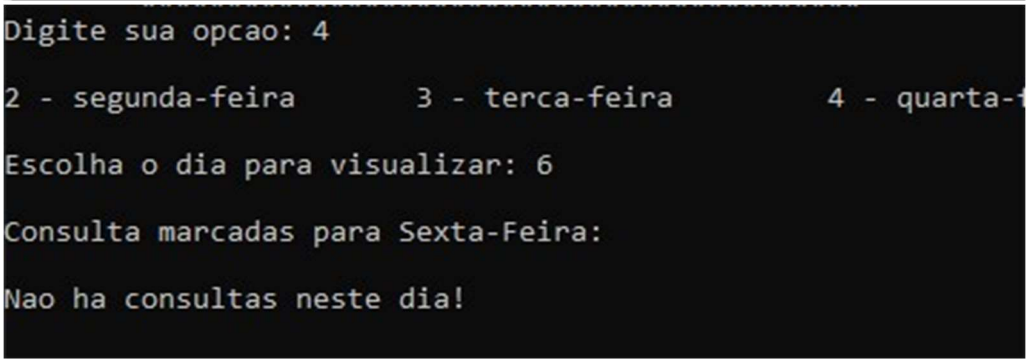
```
        ## 0 - Sair
        ############################
Digite sua opcao: 3
Digite o codigo da consulta: 62004
Consulta nao existe
```

4 - Visualizar Consultas Marcadas:

- Mostra as consultas agendadas no dia que o usuário escolher.
- Caso não existam consultas, o programa imprime mensagem informando.

Tela de execução:

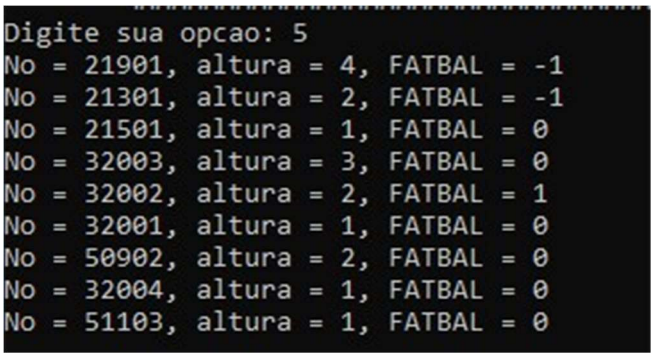




5 - Exibir Árvore:

- Imprime a estrutura da árvore. Os nós, altura, fator de balanceamento.

Tela de Execução:



6 - Sair

- Encerra o programa.

Estrutura do Algoritmo

Além dos algoritmos relativos à arvore AVL, como os de busca, inserção, impressão, rotação, foi usado alguns algoritmos das bibliotecas padrões com o objetivo de facilitar a implementação.

Funções Utilizadas

```
void pause (float segundos): Faz uma pausa de execução nas linhas do programa pelo tempo especificado. Utilizado para facilitar o uso da função rand, evitando números repetidos.
```

```
int fmax (int a, int b): Retorna o maior entre dois valores numéricos especificados
```

```
int rand (void): Quando esta função é chamada ela retorna um valor aleatório na faixa entre 0 e a constante RAND_MAX. O valor desta constante encontra-se definida no arquivo stdlib.h
```

```
void srand (unsigned int seed): inicializa a função rand com um valor “semente” de tal forma que esta semente seja um valor diferente a cada execução do programa.
```

Também criei algoritmos abstraindo a estrutura avl utilizada, afim de simplificar o programa e facilitar o entendimento.

Segue abaixo os algoritmos criados:

`int listarDia (TNo *no): exibe as consultas marcadas no dia escolhido pelo usuário. Retorna 0 se não houver consultas.`

```c
int listarSeg ( TNo * no_ptr)
{
    int aux,consultorio,hora,flag=0;;
    if ( no_ptr != NULL )
    {
        if(((no_ptr->valor)>20000)&&((no_ptr->valor)<22500)){
                    aux = no_ptr->valor;
                    consultorio = aux % 100;
                    hora = (aux/100)%100;
                    printf("Horario: %dh\n",hora);
                    printf("Consultorio: %d\n\n",consultorio);
                    flag=1;
        }

        listarSeg ( no_ptr->esq);
        listarSeg ( no_ptr->dir);
    }
    return flag;
}
```

`void buscarConsulta (AVL *avl, int Cod)`: faz a busca na árvore do COD inserido e imprime as informações da consulta.

```c
void buscarConsulta(AVL *avl, int cod){
    TNo *a = buscar(avl,cod);
    int aux;
    int dia;
    int hora;
    int consultorio;
                if(a!=NULL){
                    aux = a->valor;
                    consultorio = aux % 100;
                    hora = (aux/100)%100;
                    dia = (aux/10000);
                    printf("\nConsulta marcada para ");
                    switch(dia){
                        case 2: printf("segunda-feira\n");
                                break;
                        case 3: printf("terca-feira\n");
                                break;
                        case 4: printf("quarta-feira\n");
                                break;
                        case 5: printf("quinta-feira\n");
                                break;
                        case 6: printf("sexta-feira\n");
                                break;
                    }
                    printf("Horario: %dh\n",hora);
                    printf("Consultorio: %d",consultorio);
                }
                else {
                    printf("Consulta nao existe\n");
                }
}
```

int agendaConsulta (AVL *avl, int cod): faz o agendamento do COD gerado. Retorna 1 se a consulta foi marcada, retorna 0 se o horário estiver indisponível.

```
int agendaConsulta(AVL *avl, int cod){
    TNo *a = buscar(avl,cod);
                    if(a==NULL){
                        inserir(avl,cod);
                        printf("Seu numero de consulta: %d",cod);
                        printf("  Consulta Marcada\n");
                        return 1;
                        }
                    else{
                        printf("Seu numero de consulta: %d",cod);
                        printf("  Horario Indisponivel\n");
                        return 0;
                        }
}
```

void agendaManual (AVL *avl): utilizado pela opção 2, este algoritmo gera o COD com base no dia e horário escolhido.

```
void agendaManual(AVL *avl){
        srand(time(NULL));
        int cod;
        int dia;
        int hora;
        int aux;
        diaConsulta();
        scanf("%d",&dia);
        hour: horaConsulta();
        scanf("%d",&hora);
        if((hora<9)||(hora==12)||(hora>21)||(hora==18)){
            printf("\nHorario invalido!\n\n");
            goto hour;
        }
        int consultorio= 1;
        cod = (((dia*100)+hora)*100)+consultorio;
        aux = agendaConsultaM(avl,cod);
        if(!aux){
        do{
            consultorio++;
            cod = (((dia*100)+hora)*100)+consultorio;
            aux =agendaConsultaM(avl,cod);
            if(aux)
                break;
        }while(consultorio<4);
        }
        if(!aux)
            printf("\nHorario Indisponivel. Tente em outro horario.\n");
    }
```

`int geraConsulta(): retorna um COD aleatório.`

```
int geraConsulta(){
    srand(time(NULL));

    int dia = rand()%7;

     switch(dia){
     case 0: dia=2;
             break;
     case 1: dia=2;
             break;
    }

     int horario= rand()%22;

     while(horario<9){
         horario= rand()%22;
     }
     if(horario==12)
         horario=13;
         else if(horario==18)
             horario=19;

     int consultorio= rand()%5;

     while(consultorio==0){
         consultorio = rand()%5;
     }

     int marcacao = (dia*100)+horario;
     int consulta = (marcacao*100)+consultorio;
     return consulta;

}
```

`void n_consulta(AVL *avl, int x)`: utilizado para gerar códigos aleatoriamente com base no número pré-definido pelo usuário.

```c
void n_consultas(AVL *avl,int x){
    int i=0;
    int cod;
    int aux;
    int aux2 = 0;
    while(i<x){
        cod = geraConsulta();
        i++;
        aux = agendaConsulta(avl,cod);
        aux2=aux2+aux;
        pause(1.0);
    }

    printf("%d\n",aux2);
    printf("---------");

}
```

Corpo da Função main()

```c
int main(){

    AVL *avl1 = init_avl();
    int op;
    menu();
    dig: printf("Digite sua opcao: ");
    scanf("%d",&op);
    if((op<1)||(op>6))
        goto dig;
    int cod;
    int aux, aux2;
do{

    switch(op){
        case 1: printf("Digite o numero de consultas a ser gerado: ");
                scanf("%d",&aux);
                n_consultas(avl1,aux);
                pause(2);
                printf("\n");
                menu();
                printf("Digite sua opcao: ");
                break;
        case 2: agendaManual(avl1);
                pause(2);
                printf("\n");
                menu();
                printf("Digite sua opcao: ");
                break;
        case 3: printf("Digite o codigo da consulta: ");
                scanf("%d",&cod);
                buscarConsulta(avl1,cod);
                pause(2);
                printf("\n");
                menu();
                printf("Digite sua opcao: ");
                break;

        case 4: visualizaDia();
                scanf("%d",&aux);
                switch(aux){
                    case 2: printf("\nConsulta marcadas para Segunda-Feira:\n\n");
                            aux2=listarSeg(avl1->raiz);
                            if(!aux2)
                                printf("Nao ha consultas neste dia!\n\n");
                            break;
                    case 3: printf("\nConsulta marcadas para Terca-Feira:\n\n");
                            aux2=listarTerc(avl1->raiz);
                            if(!aux2)
                                printf("Nao ha consultas neste dia!\n\n");
                            break;
                    case 4: printf("\nConsulta marcadas para Quarta-Feira:\n\n");
                            aux2=listarQuart(avl1->raiz);
                            if(!aux2)
                                printf("Nao ha consultas neste dia!\n\n");
                            break;
                    case 5: printf("\nConsulta marcadas para Quinta-Feira:\n\n");
                            aux2=listarQuint(avl1->raiz);
                            if(!aux2)
                                printf("Nao ha consultas neste dia!\n\n");
                            break;
                    case 6: printf("\nConsulta marcadas para Sexta-Feira:\n\n");
                            aux2=listarSext(avl1->raiz);
                            if(!aux2)
                                printf("Nao ha consultas neste dia!\n\n");
                            break;

                }
```

```
                    pause(2);
                    printf("\n");
                    menu();
                    printf("Digite sua opcao: ");
                    break;
            case 5: listar(avl1->raiz);
                    pause(2);
                    printf("\n");
                    menu();
                    printf("Digite sua opcao: ");
                    break;

        }

        scanf("%d",&op);
        if((op<1)||(op>6))
            goto dig;
    }while(op!=6);
    printf("\nFIM DO PROGRAMA!\n");
    return 0;
    }

```

Referências Bibliográficas

1. https://www.vivaolinux.com.br/script/Arvore-binaria-AVL
2. https://forum.imasters.com.br/topic/453862-resolvido%C2%A0arvores-avl-em-c/
3. https://github.com/Starless2001/AVL_Tree/blob/master/AVLTree/avl.c
4. http://linguagemc.com.br/valores-aleatorios-em-c-com-a-funcao-rand/
5. https://pt.stackoverflow.com/questions/275206/%C3%89-poss%C3%ADvel-concatenar-n%C3%BAmeros-do-tipo-int
6. https://pt.wikipedia.org/wiki/%C3%81rvore_AVL
7. http://www2.ic.uff.br/~fabio/Aula-AVL-1.pdf
8. https://www.vivaolinux.com.br/script/Funcao-Temporizador-em-C
9. https://docs.microsoft.com/pt-br/cpp/c-runtime-library/reference/fmax-fmaxf-fmaxl?view=vs-2019
10. https://www.cprogressivo.net/2013/03/Como-gerar-numeros-aleatorios-em-C-com-a-rand-srand-e-seed.html