Alunos :

1 - Cristhian Mafalda (NºUSP: 9312877)
GitHub username: cristhianmafalda

2 - Sarah Tatsuta Galassi (NºUSP: 9311650)
GitHub username: sarahtgalassi

3 - Fábio Assis de Campos Júnior (NºUSP: 9312731)
GitHub username: fabioassisjr

Trabalho 2 - LPA

1-)Informações Gerais dos Programas

1-1) Introdução

Nesse trabalho, será estudada a ideologia das árvores binárias aplicadas em programação computacional. Com isso, serão desenvolvidos 3 algoritmos em linguagem C que exploram as aplicações da árvore. No primeiro, as operações básicas de inserção, remoção, busca e impressão serão feitas em uma árvore simples. No segundo, uma árvore genealógica será criada e por último, o terceiro programa será um conversor de equações em notação polonesa para outras notações.

1-2) Ambiente e Compilador

Linguagem utilizada: C
Ambiente de desenvolvimento: Code::Blocks 13.12
Compilador: GNU GCC Compiler (padrão do Code::Blocks)
Parâmetros de Compilação: padrão do ambiente
Bibliotecas utilizadas:

- stdio.h (Todos)
- stdlib.h (Todos)
- string.h (Apenas Árvore Genealógica)

1-3)GitHub

No repositório do GitHub, encontram-se :

- códigos fonte (.c) de todos os programas,
- arquivos executáveis (.exe),
- este README.txt,
- relatório em PDF,
- bibliotecas utilizadas em arquivos header(.h),
- entradas-exemplo(.txt)
- imagens das árvores correspondentes(PNG file).

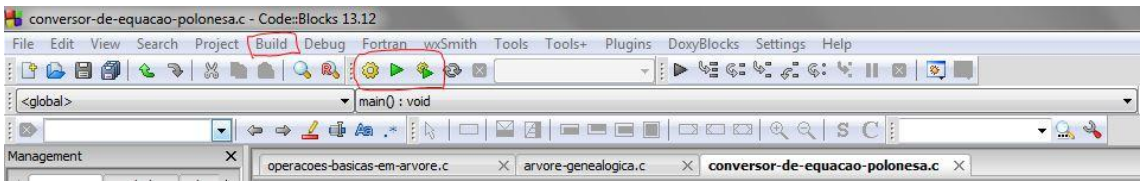
1-4)Compilação

Utilizando Code::Blocks ,o programa deve ser compilado de maneira usual, pelas ferramentas padrão do software. Dentro do software, abrir o (programa).c, a ferramenta de compilação se encontra no menu Build>Build.

Depois de compilado, o programa deve ser executado no menu Build>Run. Softwares como DEVC++, entre outros apresentam funções semelhantes como Compilar e Executar, por exemplo e podem ser usados.

Todas a compilações descritas usam GNU GCC para Windows.

Todos os programas possuem um getch() no final da execução para segurar a tela e possibilitar a visualização dos elementos de saída.



2-) Operações-Básicas-em-Árvore

2-1) Função

Operações-Básicas-em-Árvore é um programa em C onde podemos criar uma árvore binária de busca e fazer com ela as operações básicas: inserção de um novo elemento, remoção de um elemento, busca, impressão (pós ordem, em ordem, pré ordem, labelled bracketing).

2-2) Entrada

A entrada do programa deve ser constituída de um conjunto de números inteiros. O primeiro valor será a quantidade inicial(n) de valores que serão inseridos. Após isso, digite essa quantidade de inteiros do modo como deseja. Abaixo temos o exemplo usando a entrada padrão “entrada-op-1”, presente no repositório, podemos ver o cenário do programa com o menu e uma ilustração da árvore digitada.

ATENÇÃO: valores incoerentes de (n) originarão um aviso do programa e um novo valor será solicitado.

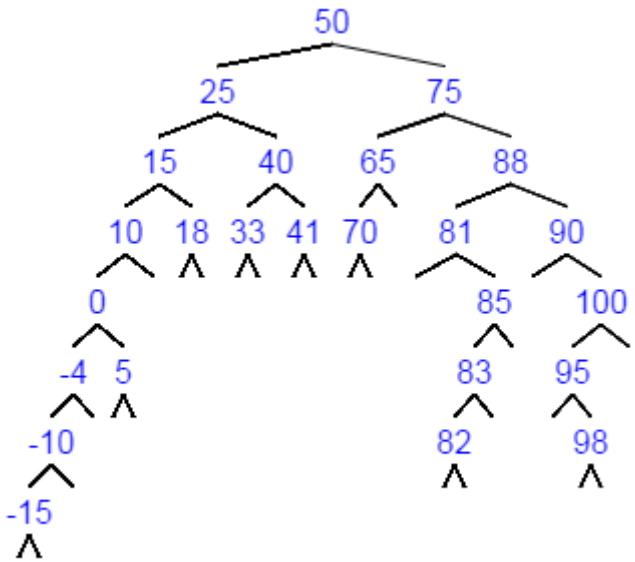
ATENÇÃO: caso sejam inseridos mais valores que (n), o programa não considera os valores de posição [n] em diante.

```
Digite a quantidade inicial de valores a ser digitado.
Em seguida, insira os valores.

25
50 25 75 15 40 65 88 10 18 33 41 70 81 90 0 -4 5 85 -10 83 -15 82 100 95 98

Digite o número da opção que deseja realizar com sua árvore:
Opção 1: Inserção
Opção 2: Remoção
Opção 3: Busca
Opção 4: Impressão Pré Ordem
Opção 5: Impressão Em Ordem
Opção 6: Impressão Pós Ordem
Opção 7: Impressão Labelled Bracketing
Opção 8: Encerrar o Programa

Opção escolhida:
```



2-3) Execução

Após a entrada, será exibido um menu contendo todas essas operações que o programa realiza. Para realizar uma operação desejada, digite o número correspondente à mesma. Sendo eles:

1. Inserção de um novo valor

   Nessa opção, assim como na entrada, será solicitado um valor (n) que será a quantidade de valores a serem adicionado e, em seguida, os (n) valores.

2. Remoção de um valor

   Ao selecionar essa opção, será requerido o valor que se pretende remover, basta digitá-lo e uma mensagem confirma a remoção. Se todos os valores da árvore forem removidos, o programa solicita um novo valor para ser a raiz, visto que não há sentido buscar, remover, ou imprimir algo de uma lista sem valores.

3. Busca

   Ao selecionar essa opção, será requerido o valor que se pretende buscar, basta digitá-lo e uma mensagem informa se o valor existe ou não na árvore.

4. Impressão em Pré-Ordem
   Nesse formato de impressão, temos que a chave é impressa antes dos seus ponteiros.

5. Impressão em Em-Ordem
   Nessa impressão, temos o ponteiro da esquerda impresso, em seguida a chave, e depois o da direita, o que gera uma impressão em ordem crescente de chaves.

6. Impressão em Pós Ordem
   A impressão desse tipo imprime ambos ponteiros antes da chave.

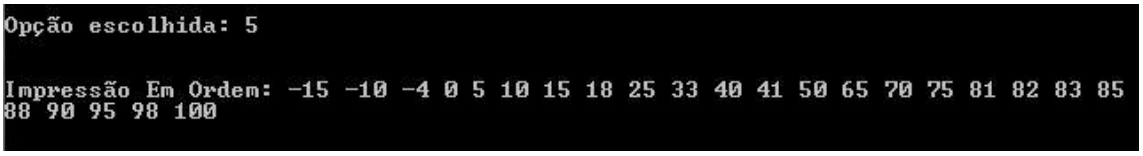
7. Impressão em Labelled-Bracketing
   Na última opção de impressão, temos esse tipo, onde cada nó é representado por um par de colchetes com sua chave e seus filhos dentro.

8. Encerrar o Programa
   Se escolhida, o programa encerra, do contrário ele realiza a função e reabre o menu para escolha de uma nova opção.

ATENÇÃO: valores incoerentes de (opção) originarão um aviso do programa e um novo valor será solicitado.

2-4) Saída

A saída padrão do programa depende da função escolhida, podendo ser um conjunto de inteiros no caso das impressões, ou apenas uma mensagem de confirmação nos demais. Exemplo de saída para a entrada usada acima para a opção de impressão em ordem da árvore:



3-) Árvore-Genealógica

3-1) Função

Árvore-Genealógica é um programa que constrói uma árvore de nomes e relações familiares , possibilitando visualizar os membros por geração, os antepassados de uma pessoa, impressão em labelled bracketing e cálculo do grau de parentesco entre duas pessoas.

3-2) Entrada

O primeiro valor deve ser um inteiro referente a quantidade(n) de conjuntos q serão colocados, que deve ser obrigatoriamente maior que zero. A seguir, deve-se inserir mais (n) conjuntos de nomes no formato (filho/pai/mãe), indica-se que essa entrada seja feita com um conjunto por linha para evitar conflitos no programa. Abaixo temos o exemplo usando a entrada padrão “entrada-gen-2”, presente no repositório, podemos ver o cenário do programa com o menu e uma ilustração da árvore digitada.

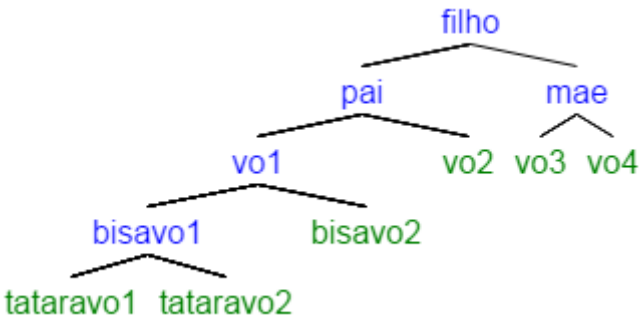
ATENÇÃO: valores incoerentes de (n) originarão um aviso do programa e um novo valor será solicitado.

```
Digite a quantidade inicial de conjuntos(filho/pai/mãe) a serem digitado
Em seguida, digite os conjuntos

5
filho pai mae
pai vo1 vo2
mae vo3 vo4
vo1 bisavo1 bisavo2
bisavo1 tataravo1 tataravo2

Digite o número da opção que deseja realizar com sua árvore genealógica:
Opção 1: Impressão dos Membros por Geração
Opção 2: Impressão dos Antepassados de uma Pessoa
Opção 3: Impressão Labelled Bracketing
Opção 4: Cálculo do Grau de Parentesco
Opção 5: Encerrar o Programa

Opção escolhida:
```



3-3) Execução

Após a entrada, será exibido um menu contendo todas essas operações que o programa realiza. Para realizar uma operação desejada, digite o número correspondente à mesma. Sendo eles:

1. Impressão dos membros por geração

   Nessa opção, teremos uma saída com um conjunto de nomes em diferentes linhas, cada qual referente a uma geração. A primeira linha refere-se a geração do primeiro elemento, a segunda, dos pais desse, e assim por diante.

2. Impressão dos antepassados

   Ao selecionar essa opção, será impressa uma linha contendo todos os nomes dos antepassados da pessoa solicitada.

3. Impressão em Labelled-Bracketing

   Nessa impressão, teremos um conjunto de chaves indicando os nós da árvore, dentre os quais cada chave possui dentro de si o nome da pessoa e as chaves de seus antepassados (caso existam na árvore).

4. Cálculo do grau de parentesco

   Digite as duas pessoas entre as quais deseja saber o grau de parentesco e o programa retorna o mesmo, sendo zero o grau que indica duas pessoas que não são parentes entre si.

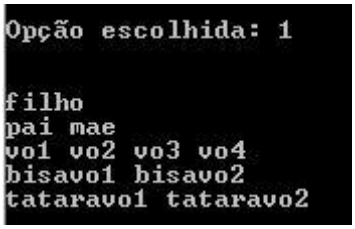
5. Encerrar o Programa

   Se escolhida, o programa encerra, do contrário ele realiza a função e reabre o menu para escolha de uma nova opção.

ATENÇÃO: valores incoerentes de (opção) originarão um aviso do programa e um novo valor será solicitado.

3-4) Saída

A saída padrão depende da opção será a impressão de um conjunto de nomes organizados, uma mensagem, ou um valor. Exemplo de saída para a entrada usada acima para a opção de imprimir membros por geração:



4-)Conversor-de-Equação-Polonesa

4-1) Função

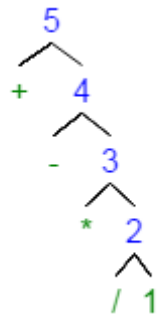
Conversor-de-Equação-Polonesa é um programa que recebe uma equação em notação polonesa e converta a mesma para notação polonesa reversa(RPN) e para notação infixa.

4-2) Entrada

A entrada deve ser composta por (n)operadores , sendo eles { / * - + }, seguidos de (n+1) números inteiros ordenados conforme a notação polonesa. Abaixo temos o exemplo usando a entrada padrão “entrada-con-1”, presente no repositório, podemos ver o cenário do programa e uma ilustração da árvore digitada.

ATENÇÃO: como todas as variáveis são lidas como char, não existe a possibilidade de entramos com números fora do intervalo de inteiros [0,9], visto que, por exemplo, o programa irá ler ‘1’ e ‘0’ onde teria que ler ‘10’. Portanto, para adequação ao algoritmo, deve-se respeitar esse fator.

```
Digite a equacao
+-*/12345
```



4-3) Saída

A saída será composta pelas duas conversões da equação inicial, tendo o tipo delas informado na frente da mesma. Exemplo de saída para a entrada usada acima:

```
RPN: 1 2 / 3 * 4 - 5 +
Infixa: ( ( ( ( 1 / 2 ) * 3 ) - 4 ) + 5 )
```

5-) Outras Informações

Existem limitações definidas para alguns programas, informadas nos parágrafos iniciados com “ATENÇÃO”, sendo a mais relevante a do terceiro programa referente ao intervalo de inteiros causados pelo uso de char. Embora outras alternativas foram testadas, elas não solucionaram esse problema e ainda causavam erros lógicos no algoritmo, logo, a funcionalidade foi mantida em detrimento do intervalo.

Portanto, os programas foram concluídos cumprindo seu objetivo inicial e o estudo das aplicações de árvore binária foram feitos com sucesso.