

**INSTITUTO POLITÉCNICO DA GUARDA**

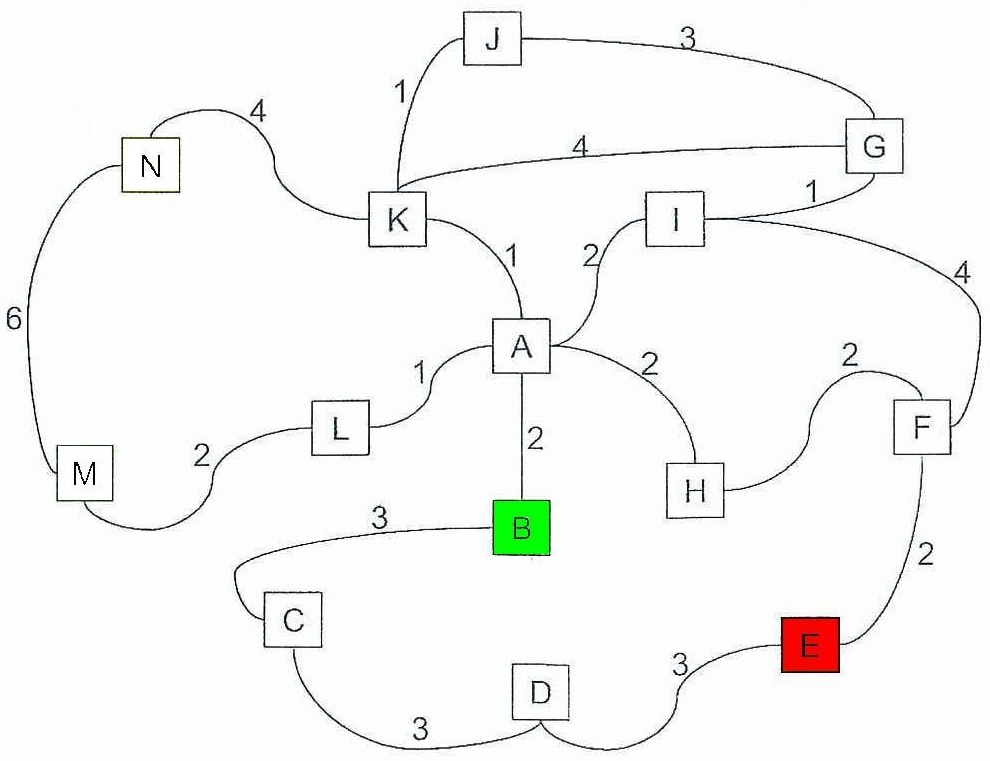
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Procura Cega - Implementação em C

3º Trabalho Obrigatório

|  |  |
| --- | --- |
| **Curso** | Licenciatura Engenharia Informática |
| **Unidade Curricular** | Inteligência Artificial |
| **Ano Letivo** | 2020/2021 |
| **Docente** | Celestino Gonçalves |
| **Data** | 19/02/2021 |
| **Aluno** | Vagner Bom Jesus nº 1701172 |

**Implementação do Algoritmo de Procura Cega em Largura Primeira**



**Codigo implementado na Linguagem C**

/\*

\* PROCURA CEGA, EM LARGURA PRIMEIRO - INTELIGÊNCIA ARTIFICIAL

\*

\* Vagner Bom Jesus, Nº 1701172

\*

\* 19/0/2021

\*

\*

\*/

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <string.h>

void pausa() {

puts("\nPremir 'Enter' para continuar.");

scanf("%\*c");

getchar();

}

//

// estrutura de nós

//

typedef struct

{

char nomeNo[10];

char nomeCidade[10];

int distancia;

} CIDADES;

//

// estrutura da solução: percurso e distância

//

typedef struct {

char nome[10];

int dist;

int repetido;

} SOLUS;

//

// função main()

//

int main()

{

setlocale(LC\_ALL, "Portuguese");

//

// preencher a estrutura a partir de um ficheiro de texto

//

CIDADES nos[46];

FILE \*f;

if ((f = fopen("dados.txt", "rt")) == NULL)

{

puts("Erro ao abrir ficheiro!");

exit(1);

}

int i = 0, numeroElementos = 0;

char linha[100];

while (fgets(linha, 100, f))

{

sscanf(linha, "%s %s %d", nos[i].nomeNo, nos[i].nomeCidade, &nos[i].distancia);

numeroElementos = i;

++i;

}

fclose(f);

puts("");

//

// lista de nós

//

char listaNos[14][10] = { "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N"};

puts("Por exemplo ' B ' (nó de origem 1) e ' E ' (nó de destino 4).\n");

// while (1 == 1) {

i = 0;

puts("Lista de nós:\n");

for (i = 0; i < 14; ++i)

{

printf("% d) %s\n", i, listaNos[i]);

}

puts("");

int noOrigem, noDestino;

do

{

puts("Escolher nó de origem [0 a 13]:");

scanf("%d", &noOrigem);

puts("Escolher nó de destino [0 a 13]:");

scanf("%d", &noDestino);

if (noOrigem == noDestino)

{

puts("O nó de origem deve ser diferente do nó de destino!\n");

}

else if (noOrigem < 0 || noDestino < 0 || noOrigem>13 || noDestino>13)

{

puts("Escolher nós dentro do intervalo [0, 13]!\n");

}

} while (noOrigem < 0 || noDestino < 0 || noOrigem>13 || noDestino>13 || noDestino == noOrigem);

// gerar árvore de pesquisa

//

int j = 0, k = 0;

char proximoNo[10];

strcpy(proximoNo, listaNos[noOrigem]);

char expansaoNos[5000][10];

do

{

printf("\n\n\nIteração [%d]: %s\n", k + 1, proximoNo);

for (i = 0; i <= numeroElementos; ++i)

{

if (!strcmp(proximoNo, nos[i].nomeNo)) //strcmp() retorna 0 se há correspondência

{

strcpy(expansaoNos[j], nos[i].nomeCidade);

printf("\nNós: %s", expansaoNos[j]);

++j;

}

}

strcpy(proximoNo, expansaoNos[k]);

++k;

if (!strcmp(proximoNo, listaNos[noDestino])) {

printf("\n\n\nNó de destino encontrado.\nIteração [%d]: %s\n", k + 1, proximoNo);

break;

}

} while (strcmp(proximoNo, listaNos[noDestino]));

//

// apresentar a solução

//

SOLUS solucao[100];

char noAnterior[10];

strcpy(noAnterior, listaNos[noDestino]);

strcpy(solucao[0].nome, listaNos[noDestino]);

solucao[0].dist = 0;

j = 1;

for (i = 0; i < 100; ++i) {

solucao[i].repetido = 0; // inicialização da variável

}

do

{

for (i = 0; i <= numeroElementos; ++i)

{

if ((!strcmp(noAnterior, nos[i].nomeCidade)) && (solucao[j - 1].repetido == 0))

{

strcpy(solucao[j].nome, nos[i].nomeNo);

strcpy(noAnterior, nos[i].nomeNo);

solucao[j].dist = nos[i].distancia;

solucao[j - 1].repetido = 1;

++j;

}

if (!strcmp(noAnterior, listaNos[noOrigem])) {

break;

}

}

} while (strcmp(noAnterior, listaNos[noOrigem]));

puts("\n\nSolução:");

for (i = j - 1; i >= 0; --i) {

if (i == 0) {

printf("%s", solucao[i].nome);

break;

}

printf("%s -> (%d) ", solucao[i].nome, solucao[i].dist);

}

puts("\n");

int distTotal = 0;

puts("\nCusto da solução:");

for (i = j - 1; i >= 0; --i) {

distTotal += solucao[i].dist;

}

printf("%d (km)", distTotal);

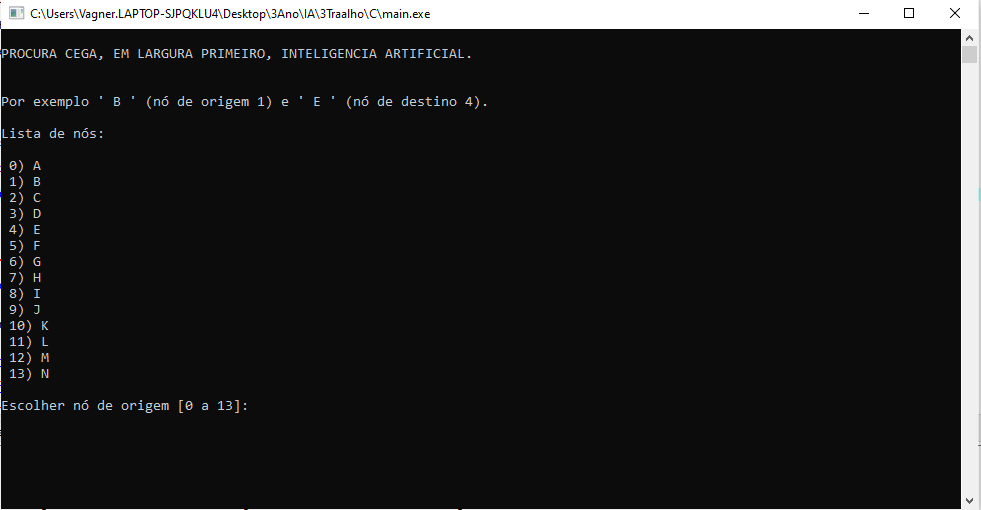
puts("\n");

pausa();

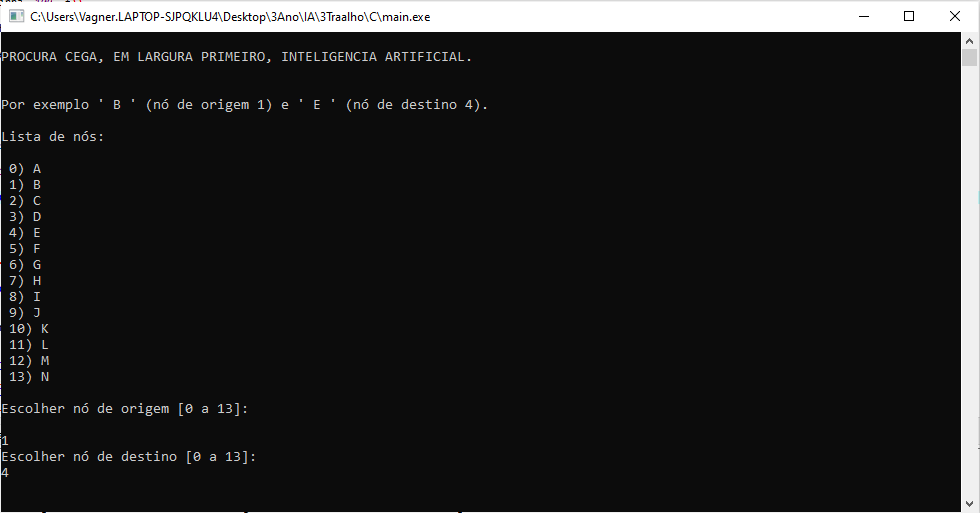
return (0);

}

**Ecrã Principal do programa**



**Ecrã escolha dos nós**



**Ecrã apresenta as Intenções, Solução e o custo da solução do programa**

