1º)Questão:

static int cnt;

void DIGRAPHdfs(Digraph G) {

Vertex v;

cnt = 0;

for (v = 0; v < G->V; v++)

lbl[v] = -1;

for (v = 0; v < G->V; v++)

if (lbl[v] == -1)

dfsR(G,v);

}

void dfsR(Digraph G, Vertex v) {

Vertex w;

lbl[v] = cnt++;

for (w = 0; w < G->V; w++)

if (G->adj[v][w] != 0)

if (lbl[w] == -1)

dfsR(G, w);

}

2º)Questão:

Vetor de lbl = 0,1,2,3,4,5,6,7,8,9

void showlbl(Digraph G) {

Vertex v;

int i;

printf ("ind: ");

for (i = 0; i < G->V; i++)

printf("%d - ",i);

printf("\n");

printf("lbl: ");

for (v = 0; v < G->V; v++)

printf("%d - ",lbl[v]);

}

4º)Questão:

Arcos existentes: 0-1,0-4,0-5,4-1,4-5,5-3,3-4

Vertices existentes: 0,1,2,3,4,5

Arco 3-4 é um arco de retorno

Arco 0-5 é um arco descendente

Arco 5-3 é um arco da arborecência

Arco 4-1 é um arco cruzado

5º)Questão:

static Vertex parnt[maxV];

static int cnt;

void DIGRAPHbfs(Digraph G, Vertex s) {

Vertex v, w;

cnt = 0;

for (v = 0; v < G->V; v++)

lbl[v] = -1;

QUEUEinit(G->V);

lbl[s] =cnt++;

parnt[s] = s;

QUEUEput(s);

while(!QUEUEempty()) {

v = QUEUEget();

for (w = 0; w < G->V; w++)

if (G->adj[v][w] == 1 && lbl[w] == -1) {

lbl[w] = cnt++;

parnt[w] = v;

QUEUEput(w);

}

}

QUEUEfree();

}

Arquivo Queue.c:

#include "stdlib.h"

#include "stdio.h"

static int maxN = 1000;

typedef Vertex Item;

Item \*q;

int inicio, fim;

void QUEUEinit(int maxN) {

q = (Item\*) malloc(maxN\*sizeof(Item));

inicio = 0;

fim = 0;

}

int QUEUEempty() {

return inicio == fim;

}

void QUEUEput(Item item) {

q[fim++] = item;

}

Item QUEUEget() {

return q[inicio++];

}

void QUEUEfree() {

free(q);

}

6º)Questão:

O codigo do lbl é o mesmo da questão 2.

Utiliza também o arquivo Queue.c

Vetor de lbl: 0,1,2,4,3,6,7,8,9,5

7°)Questão:

static Vertex parnt[maxV];

static int cnt;

void DIGRAPHbfs(Digraph G, Vertex s) {

Vertex v;

link p;

cnt = 0;

for (v = 0; v < G->V; v++)

lbl[v] = -1;

QUEUEinit(G->V);

lbl[s] =cnt++;

parnt[s] = s;

QUEUEput(s);

while(!QUEUEempty()) {

v = QUEUEget();

for (p = G->adj[v]; p != NULL; p = p->next)

if (lbl[p->w] == -1) {

lbl[p->w] = cnt++;

parnt[p->w] = v;

QUEUEput(p->w);

}

}

QUEUEfree();

}

8º)Questão:

#define INFINITO 1000

static int dist[maxV];

void DIGRAPHbfs2 (Digraph G, Vertex s) {

Vertex v, w;

for (v = 0; v < G->V; v++) {

dist[v] = INFINITO;

parnt[v] = -1;

}

QUEUEinit(G->V);

dist[s] = 0;

parnt[s] = s;

QUEUEput(s);

while (!QUEUEempty()) {

v = QUEUEget();

for (w = 0; w < G->V; w++)

if (G->adj[v][w] == 1 && dist[w] == INFINITO) {

dist[w] =dist[v] + 1;

parnt[w] = v;

QUEUEput(w);

}

}

QUEUEfree();

}

void showdist(Digraph G) {

Vertex v;

int i;

printf ("indi: ");

for (i = 0; i < G->V; i++)

printf("%d - ",i);

printf("\n");

printf("dist: ");

for (v = 0; v < G->V; v++)

printf("%d - ",dist[v]);

}

Vetor da distancia: 0,1,2,3,2,3,3,3,4,3