**Implementation of Graph Using Linked List**

#include<stdio.h>

#include<stdlib.h>

struct edge

{

int vertexIndex;

struct edge \*edgePtr;

};

struct vertex

{

int vertexKey;

struct edge \*edgePtr;

};

struct vertex graph[10];

int vertexCount = 0;

void InsertVertex(int vertexKey)

{

graph[vertexCount].vertexKey = vertexKey;

graph[vertexCount].edgePtr = NULL;

vertexCount++;

}

void insertEdge(int vertex1, int vertex2)

{

struct edge \*e,\*e1,\*e2;

e = graph[vertex1].edgePtr;

e1 = (struct edge \*)malloc(sizeof(struct edge));

e1->vertexIndex = vertex2;

e1->edgePtr = NULL;

while(e&& e->edgePtr)

{

e = e->edgePtr;

}

if(e)

e->edgePtr = e1;

else

graph[vertex1].edgePtr = e1;

e = graph[vertex2].edgePtr;

e2 = (struct edge \*)malloc(sizeof(struct edge));

e2->vertexIndex = vertex1;

e2->edgePtr = NULL;

while(e&& e->edgePtr)

{

e = e->edgePtr;

}

if(e)

e->edgePtr = e2;

else

graph[vertex2].edgePtr = e2;

}

void printGraph()

{

int i;

struct edge \*e;

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

{

printf("%d(%d)", i, graph[i].vertexKey);

e = graph[i].edgePtr;

while(e)

{

printf("->%d", e->vertexIndex);

e = e->edgePtr;

}

printf("\n");

}

}

void main()

{

InsertVertex(5);

InsertVertex(3);

InsertVertex(4);

InsertVertex(2);

InsertVertex(9);

insertEdge(0,1);

insertEdge(0,2);

insertEdge(1,3);

insertEdge(1,4);

printGraph();

}