Unidad 6: Grafos

Algoritmos y Estructuras de Datos

Operaciones
Tipos
Algoritmos

Ing. Juan Ignacio Iturriaga

Grafos - Operaciones

Acciones básicas

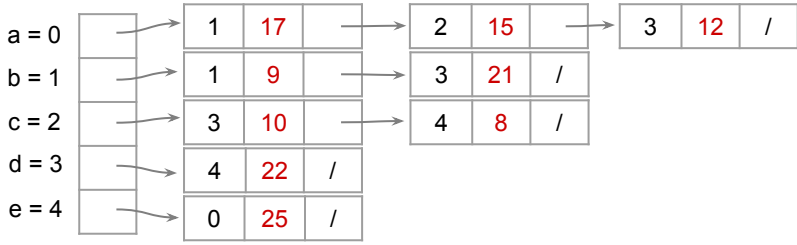
Matriz de adyacencia

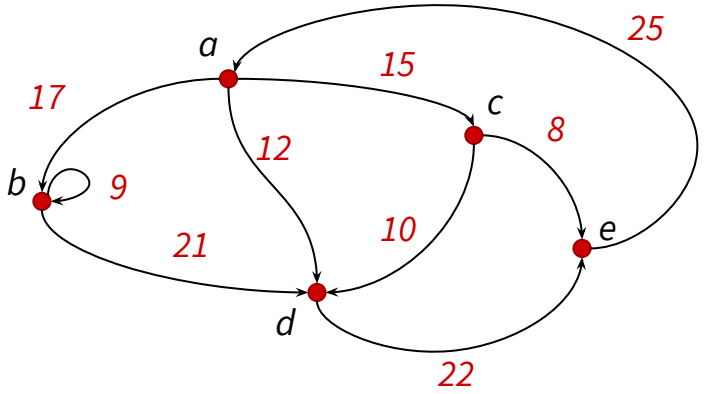
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista

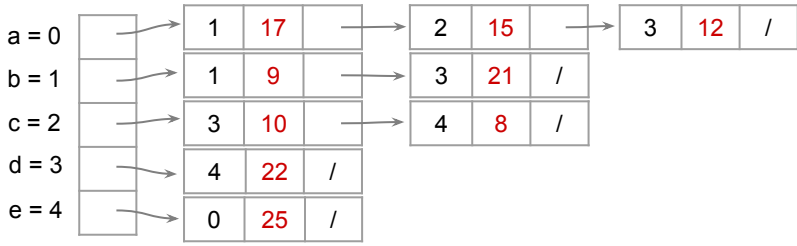
Matriz de adyacencia

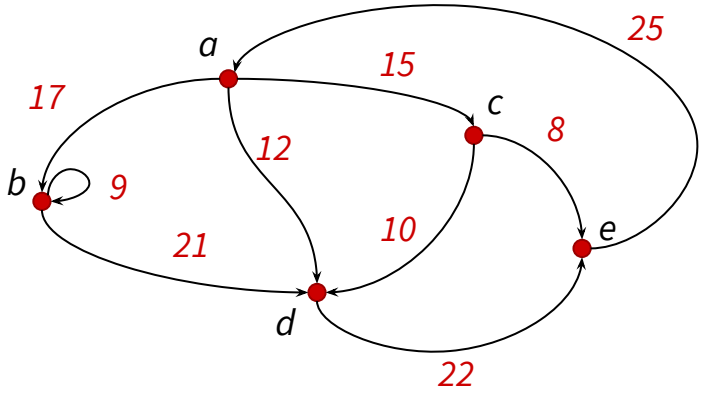
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista

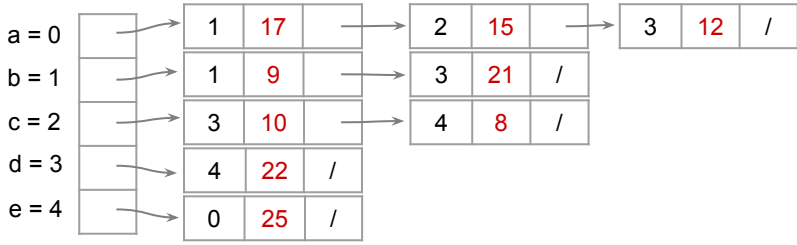
Matriz de adyacencia

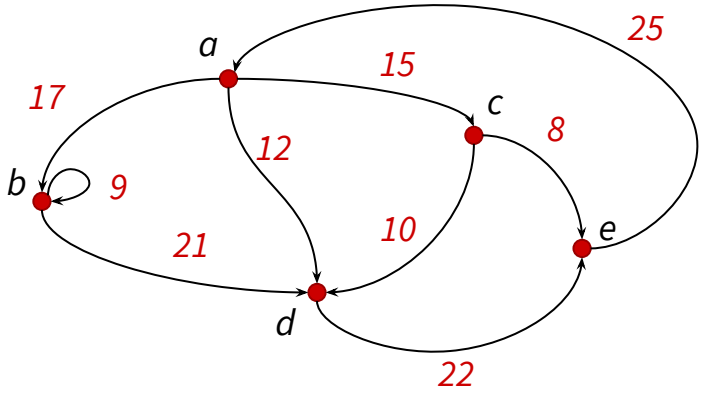
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista
- Insertar vértice

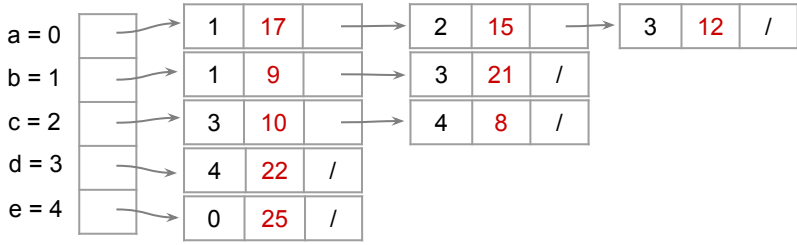
Matriz de adyacencia

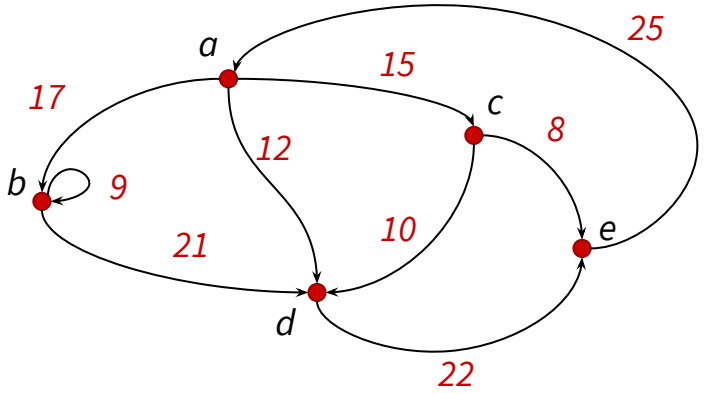
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista
- Insertar vértice
- Eliminar vértice

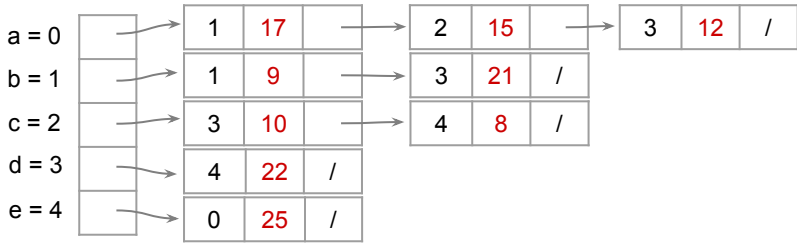
Matriz de adyacencia

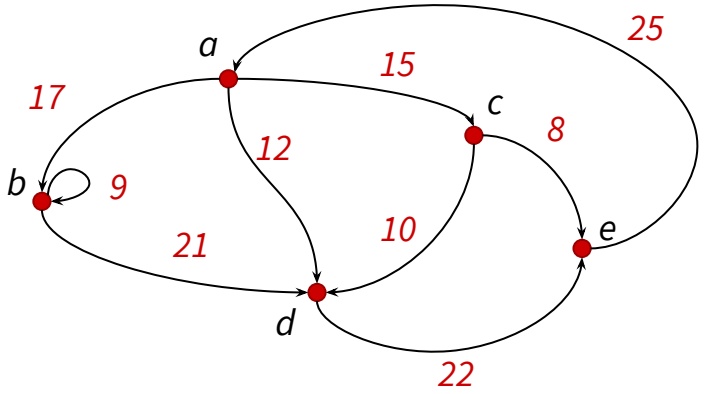
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista
- Insertar vértice
- Eliminar vértice
- Comprobar la existencia de una arista

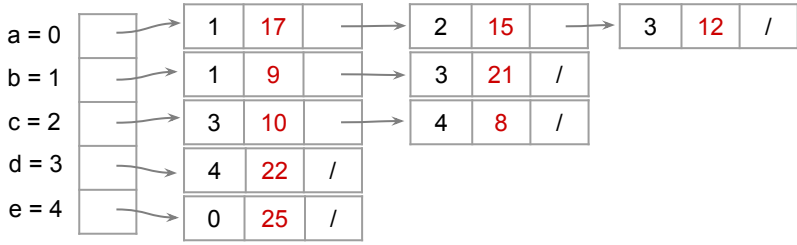
Matriz de adyacencia

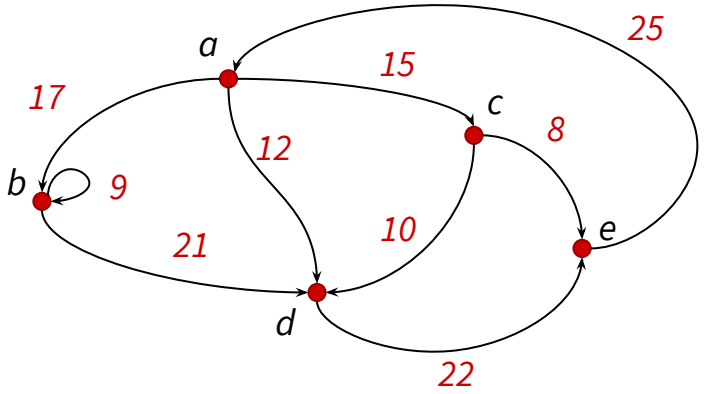
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista
- Insertar vértice
- Eliminar vértice
- Comprobar la existencia de una arista
- Verificar si un vértice es adyacente de otro

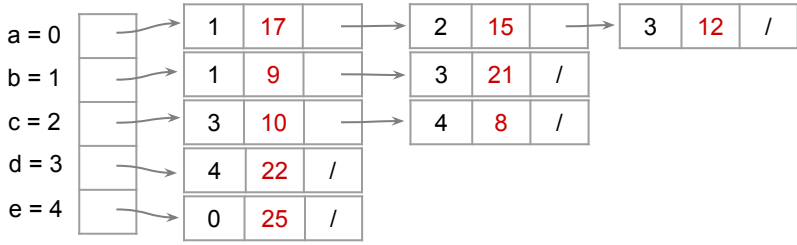
Matriz de adyacencia

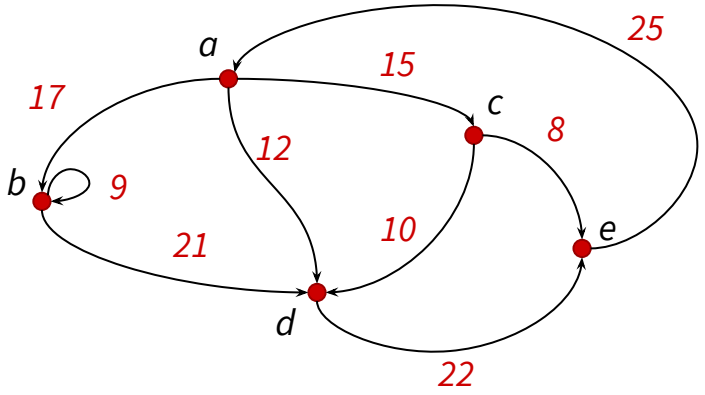
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista
- Insertar vértice
- Eliminar vértice
- Comprobar la existencia de una arista
- Verificar si un vértice es adyacente de otro
- Obtener la lista de vértices adyacentes

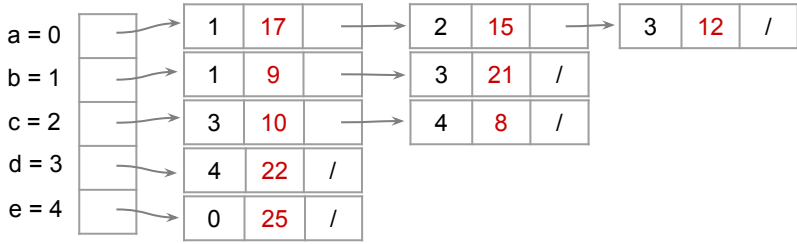
Matriz de adyacencia

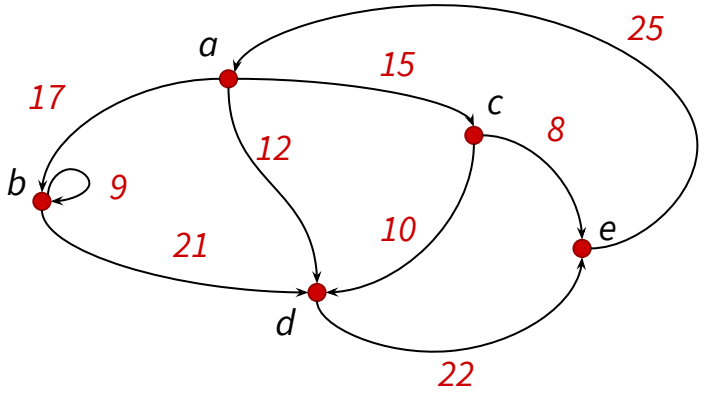
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Operaciones

Acciones básicas

¿Cómo resolver operaciones básicas?

- Insertar arista
- Eliminar arista
- Insertar vértice
- Eliminar vértice
- Comprobar la existencia de una arista
- Verificar si un vértice es adyacente de otro
- Obtener la lista de vértices adyacentes
- Obtener el costo de recorrer una arista

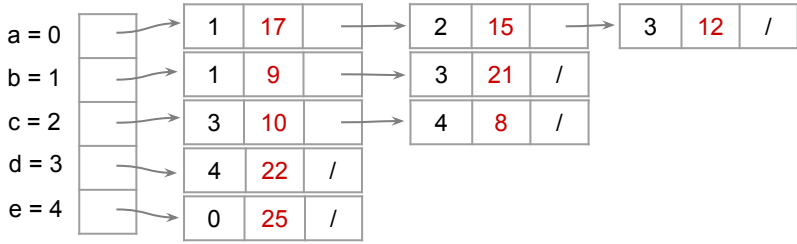
Matriz de adyacencia

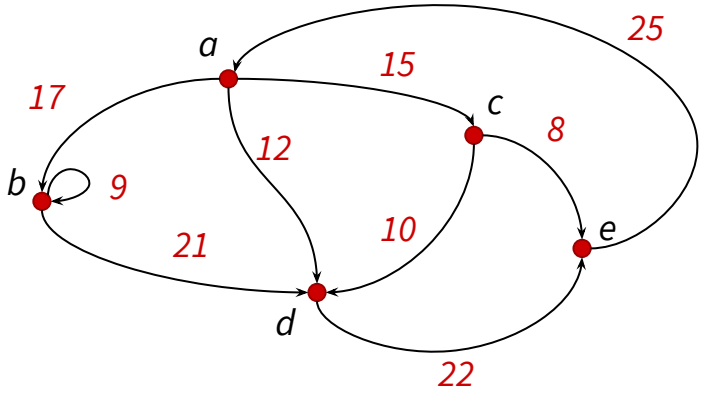
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo

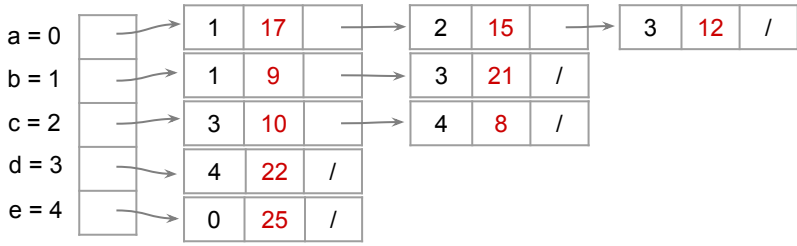
Matriz de adyacencia

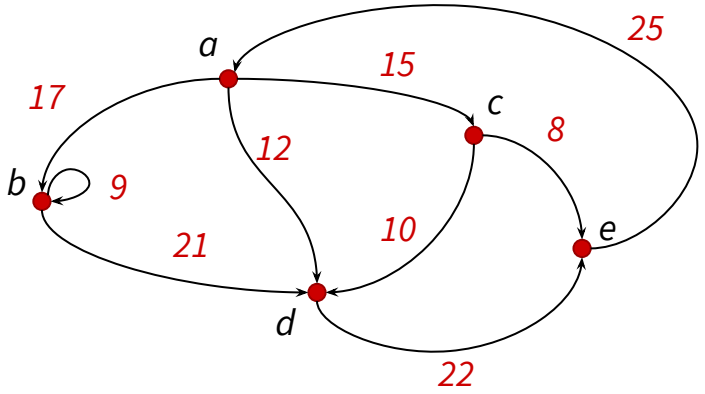
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano

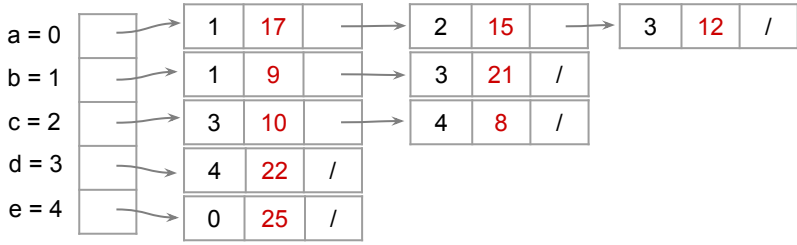
Matriz de adyacencia

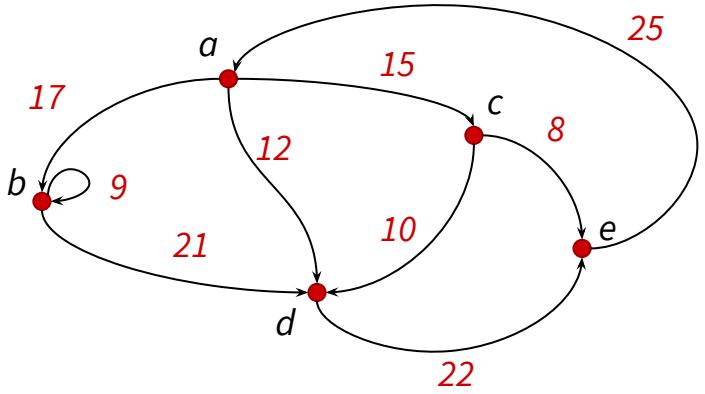
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano

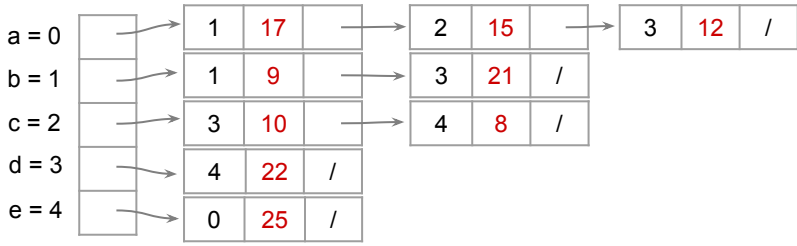
Matriz de adyacencia

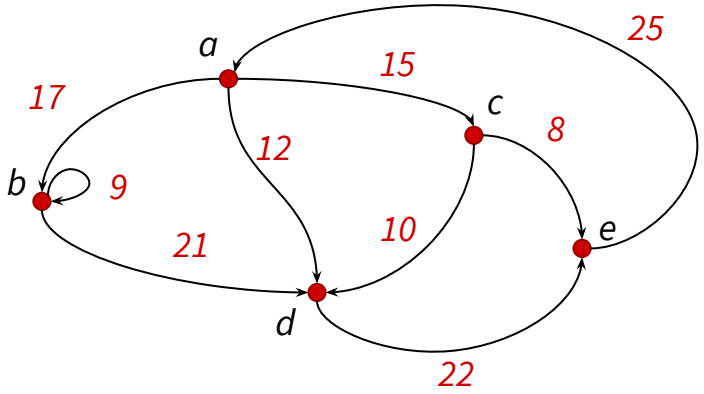
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito

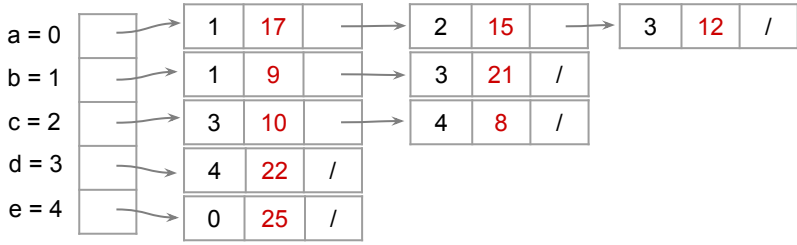
Matriz de adyacencia

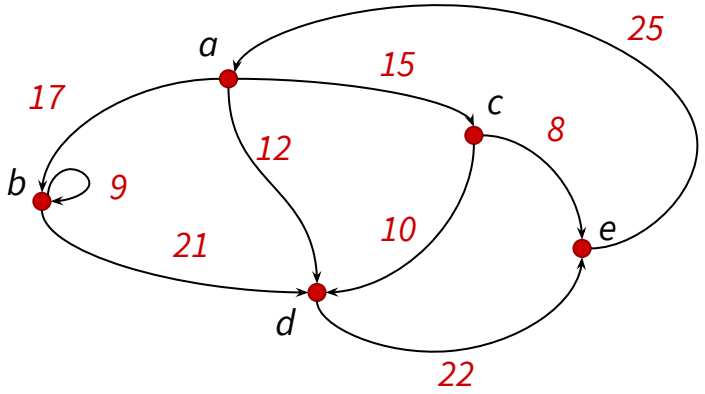
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito
- Conexo

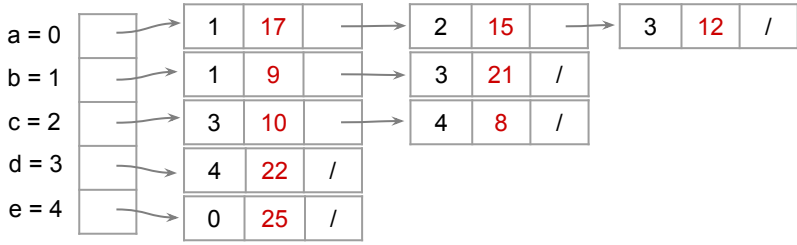
Matriz de adyacencia

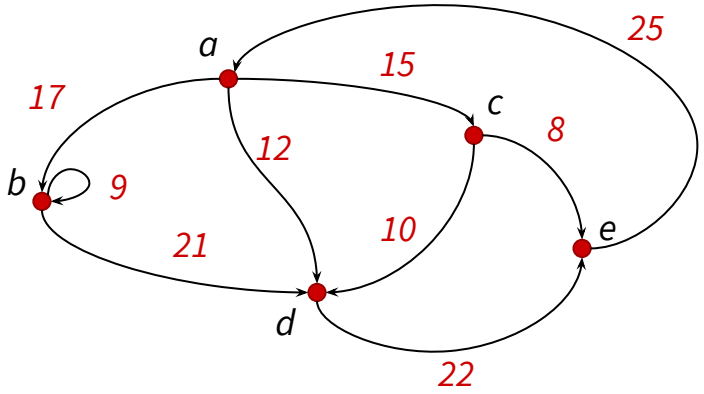
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito
- Conexo
- Orientado

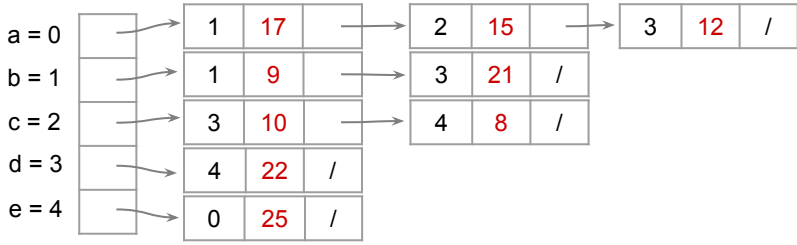
Matriz de adyacencia

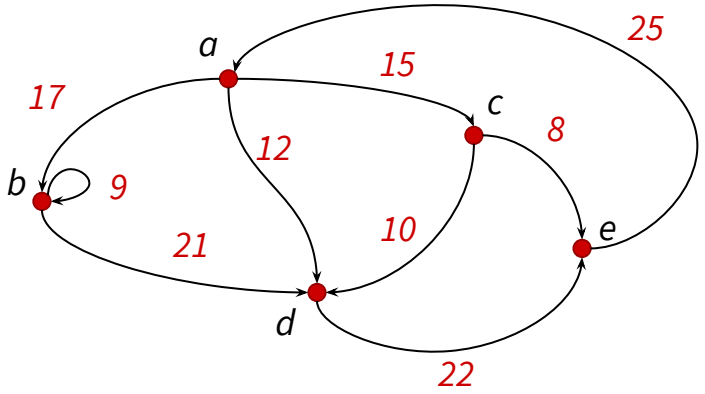
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito
- Conexo
- Orientado
- Completo

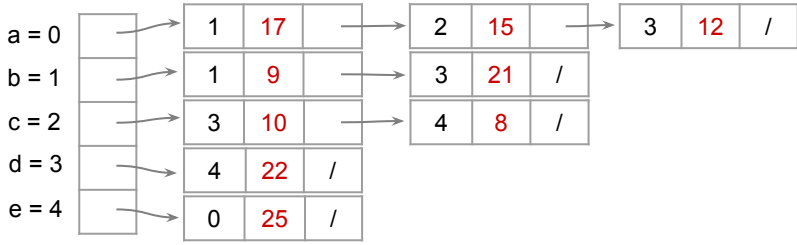
Matriz de adyacencia

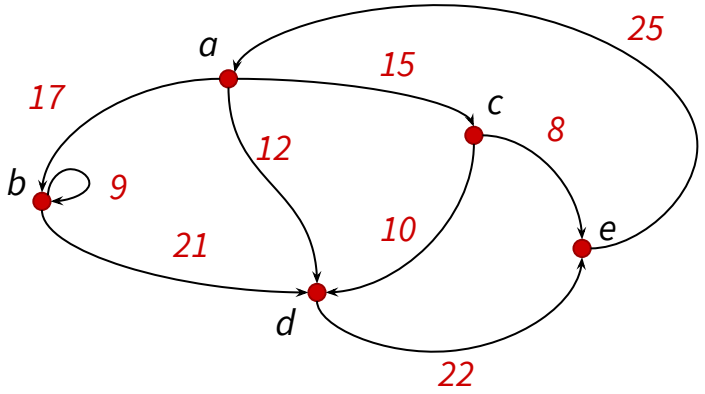
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito
- Conexo
- Orientado
- Completo
- Vacío

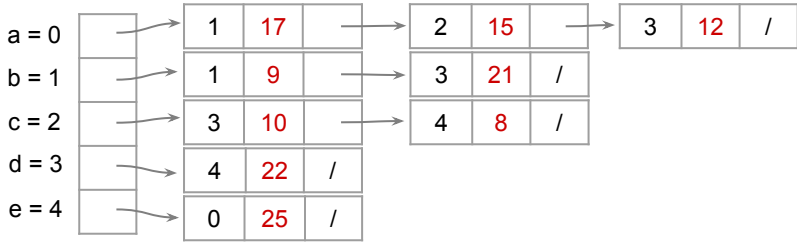
Matriz de adyacencia

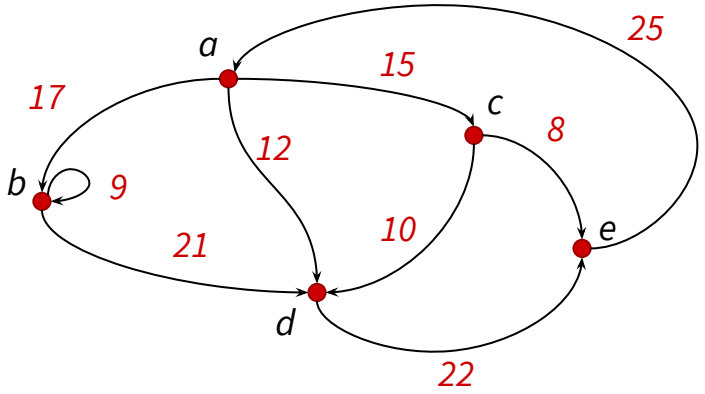
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito
- Conexo
- Orientado
- Completo
- Vacío
- Denso o Disperso

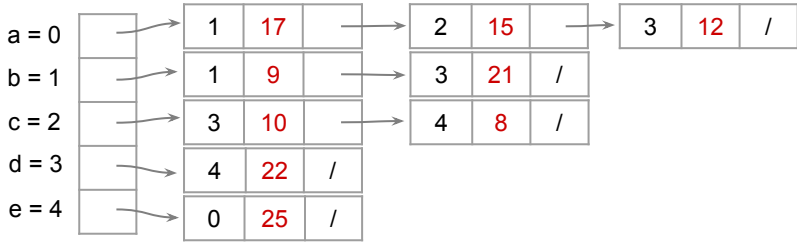
Matriz de adyacencia

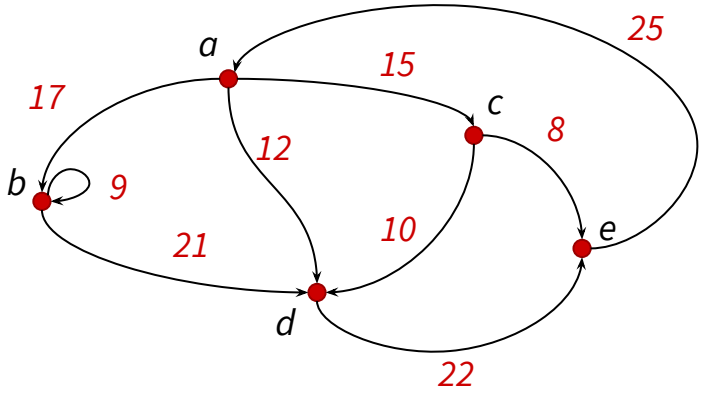
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - tipos

Clasificación de gráfos

¿Cómo saber que si un grafo es ...?

- Grafo o digrafo
- Grafo euleriano
  - Encontrar el ciclo euleriano
- Bipartito
- Conexo
- Orientado
- Completo
- Vacío
- Denso o Disperso
- Simple o no simple

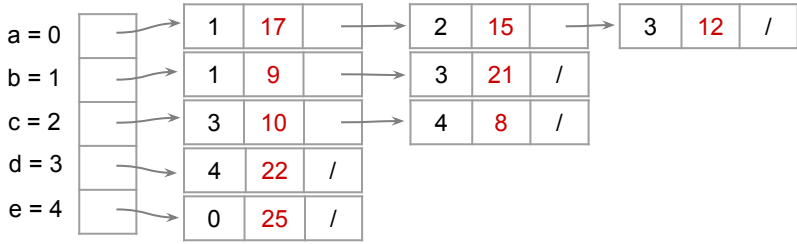
Matriz de adyacencia

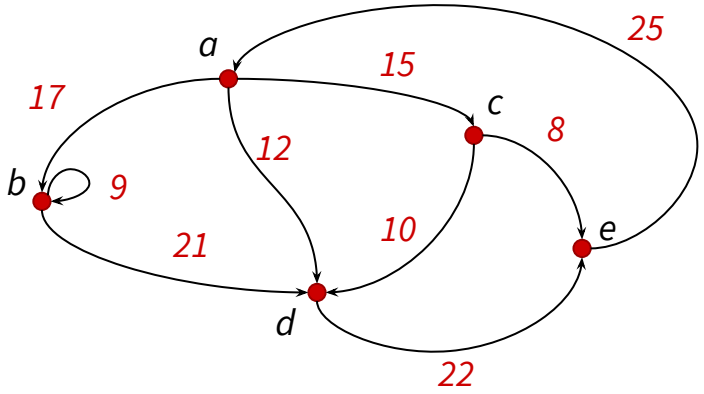
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

Recorrido en anchura (amplitud)

```
void graphBFP (graph *g, int from){
    if (!g) return;
    if (g->size <= from) return;
    int v[MAX];
    for (int i = 0; i < g->size; i++){
        v[i] = 0;
    }

    queue *q = createQ();

    v[from] = 1;
    enqueueQ(q,from);
    while (notEmptyQ(q)){
        int e = dequeueQ(q);
        for (int i = 0; i < g->size;i++){
            if ((v[i]==0) && (g->A[e][i] != 0)){
                v[i] = 1;
                enqueueQ(q,i);
            }
        }
        printf ("%s ", g->V[e]->value);
    }
    printf ("\n");
}
```

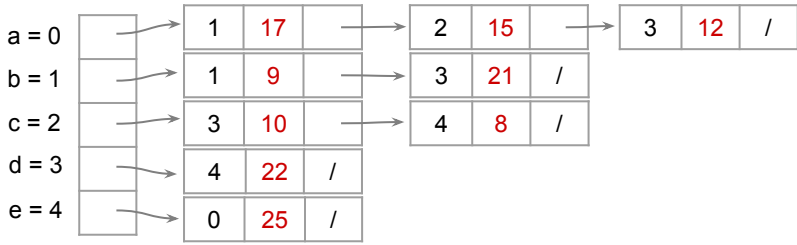
Matriz de adyacencia

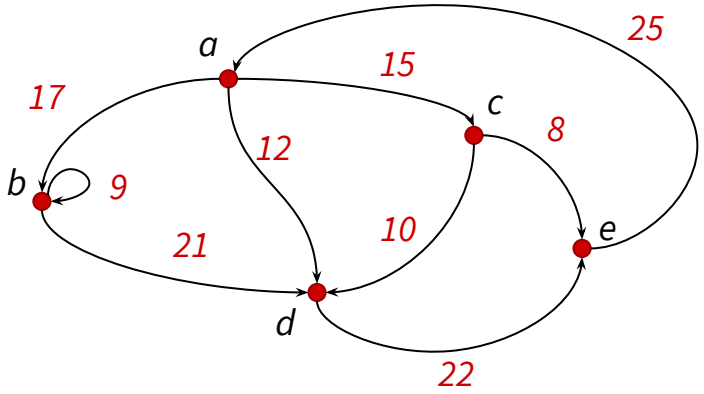
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

Una posible implementación

```
typedef struct Graph graph;
typedef struct GraphVertex vertex;

typedef struct GraphVertex {
    int index;
    char *value;
} vertex;

typedef struct Graph {
    vertex *V[MAX];
    int A[MAX][MAX];
    int size;
} graph;
```

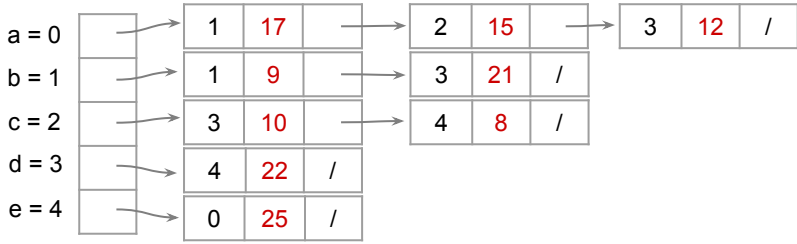
Matriz de adyacencia

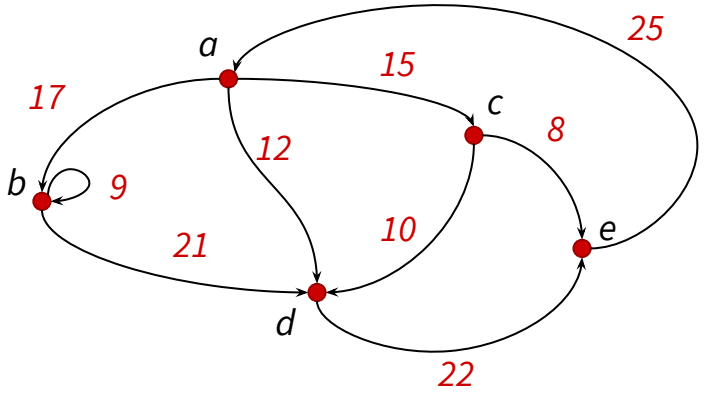
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

Recorrido en profundidad

```
void graphDFP (graph *g, int from){        //ACTUALIZADO
    if (!g) return;
    if (g->size <= from) return;

    int v[MAX];
    for (int i = 0; i < g->size; i++){
        v[i] = 0;
    }

    stack *s = createS();

    push(&s,from);
    while (!isEmptyS(s)){
        int e = pop(&s);
        if (v[e] == 0){
            for (int i = g->size - 1; i >= 0 ;--i){
                if (g->A[e][i] != 0){
                    push(&s,i);
                }
            }
            v[e] = 1;
            printf ("%s ", g->V[e]->value);
        }
    }
    printf ("\n");
}
```

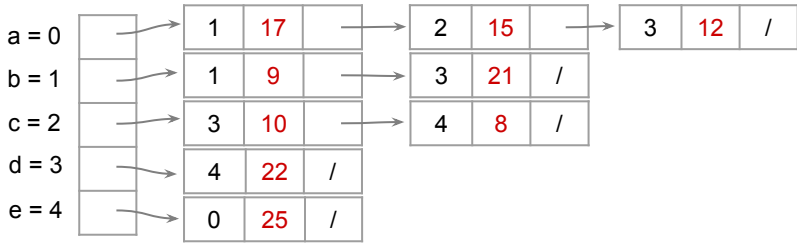
Matriz de adyacencia

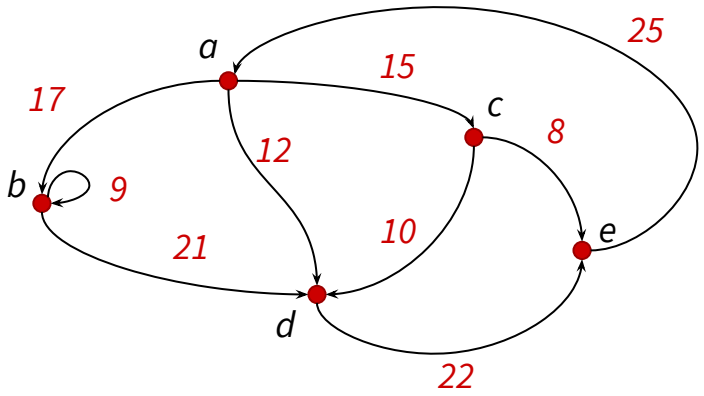
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

Árbol de cobertura en amplitud

```
ntn *graphBFTree (graph *g, int from){
    if (!g) return NULL;
    if (g->size <= from) return NULL;

    ntn *root = createNTN(from);

    int v[MAX];
    for (int i = 0; i < g->size; i++){
        v[i] = 0;
    }

    queue *q = createQ();

    v[from] = 1;

    enqueueQ(q,from);
    while (notEmptyQ(q)){
        int e = dequeueQ(q);
        for (int i = 0; i < g->size;i++){
            if ((v[i]==0) && (g->A[e][i] != 0)){
                v[i] = 1;
                insertSonValueOfNTN(root, e, i);
                enqueueQ(q,i);
            }
        }

    }
    return root;
}
```

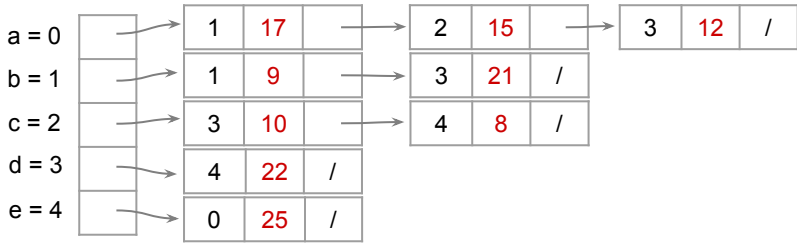
Matriz de adyacencia

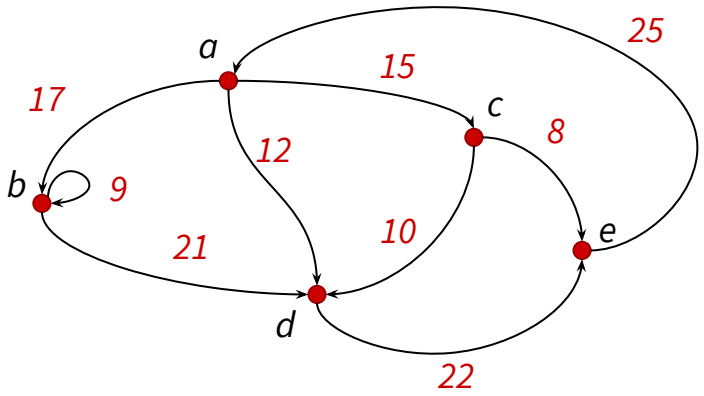
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

Árbol de cobertura en profundidad

```
void _graphDFTree (graph *g, ntn *root, int parent, int vertex, int
visited[]){
    if (!g) return;
    if (!root) return;
    if (visited[vertex] == 0){
        visited[vertex] = 1;
        insertSonValueOfNTN (root, parent, vertex);
        for (int i = 0; i < g->size; i++){
            if (g->A[vertex][i]>0){
                _graphDFTree(g,root,vertex,i,visited);
            }
        }
    }
}

ntn *graphDFTree (graph *g, int from){
    if (!g) return NULL;
    if (g->size <= from) return NULL;
    int v[MAX];
    for (int i = 0; i < g->size; i++){
        v[i] = 0;
    }
    ntn *root = createNTN(from);
    v[from] = 1;
    for (int i = 0; i < g->size; i++){
        if (g->A[from][i]>0){
                _graphDFTree(g,root,from,i,v);
        }
    }
    return root;

}
```

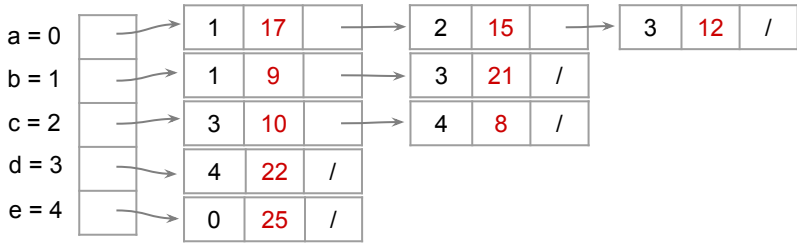
Matriz de adyacencia

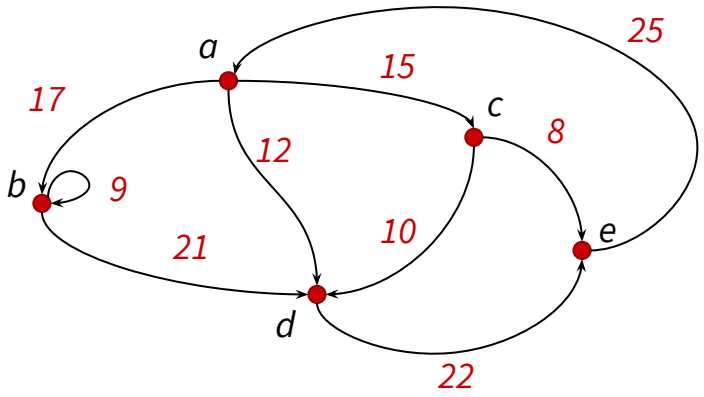
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

KRUSKAL

```
graph *kruskal(graph *g){

    if (!g) return NULL;
    graph *k = createGraph();
    int v[MAX];
    for (int i = 0; i < g->size; i++){
        v[i] = 0;
        graphAddNewVertex (k, strdup(g->V[i]->value));
    }


    int minFrom, minTo;
    while (count < (g->size-1)){
        int minCost = INF;
        for (int i = 0; i < g->size; i++){
            for (int j = i; j < g->size; j++){
                if ((v[j] == 0)&&(graphCost(g,i,j)<minCost)){
                    minCost = graphCost(g,i,j);
                    minFrom = i;
                    minTo = j;
                }
            }
        }
        graphSetArc(k, minFrom, minTo, minCost);
        graphSetArc(k, minTo, minFrom, minCost);
        v[minFrom] = 1;
        v[minTo] = 1;
    }

    return k;

}
```

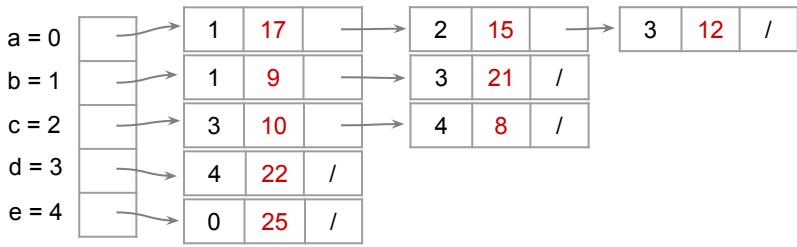
Matriz de adyacencia

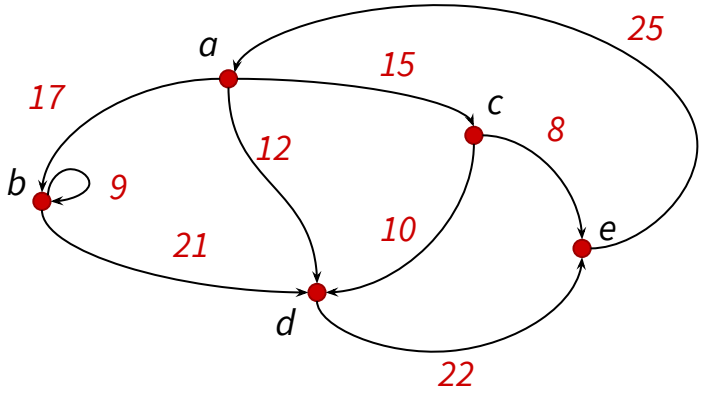
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Grafos - Algoritmos

FLOYD

```
typedef struct FloydRes fres;

typedef struct FloydRes {
    int D[MAX][MAX];
    int C[MAX][MAX];
    int count;
} fres;


void floyd(graph *g, fres *r) {
    r->count = g->size;

    for (int i = 0; i < g->size; i++) {
        for (int j = 0; j < g->size; j++) {
            r->D[i][j] = i;
            r->C[i][j] = graphCost(g, i, j);
        }
    }

    for (int k = 0; k < g->size; k++) {
        for (int i = 0; i < g->size; i++) {
            for (int j = 0; j < g->size; j++) {
                int cost = (r->C[i][k] + r->C[k][j]);
                if ((i!=j)&&(cost < r->C[i][j])){
                    r->D[i][j] = r->D[k][j];
                    r->C[i][j] = cost;
                }
            }
        }
    }

}
```

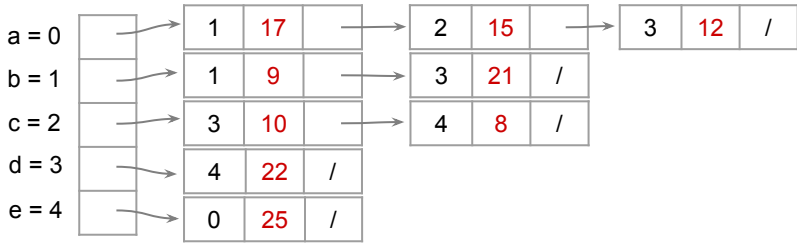
Matriz de adyacencia

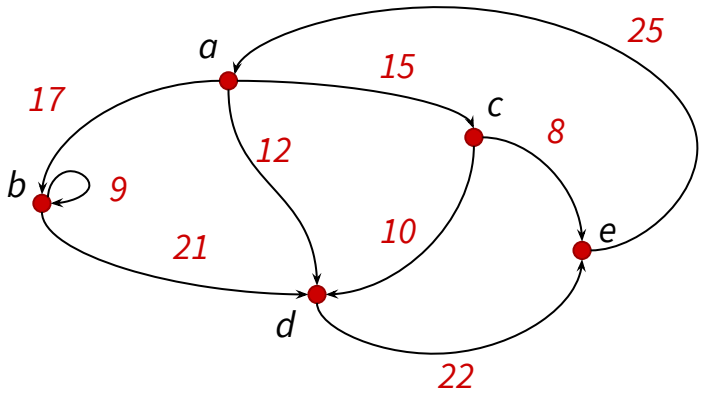
Adjacency Matrix

	0	1	2	3	4
a = 0	0	17	15	12	0
b = 1	0	9	0	21	0
c = 2	0	0	0	10	8
d = 3	0	0	0	0	22
e = 4	25	0	0	0	0

Listas de adyacencia

Adjacency List





Unidad 6: Grafos

Algoritmos y Estructuras de Datos

Operaciones
Tipos
Algoritmos

Ing. Juan Ignacio Iturriaga