	HT
0	20
1	
2	12
3	
4	
5	
6	6
7	17
8	
9	

$h(12) = 12\%10 = 2$ index in HT

$h(\underline{20}) = 20\%10 = \underline{0}$ index in HT

$h(27) = 27\%10 = \underline{7}$ → Collision happens

To resolve Collision, We have two methods:

1- Open Hashing (or Closed Addressing) → We use "chaining method" (Linked List)

2- Closed Hashing (or open Addressing)

a- Linear probing → $(u+i)\%m$

b- Quadratic probing → $(u+i^2)\%m$

c- double hashing. $H_1(k)$, $H_2(k)$

ex #1a: Key Values: KV = 3, 2, 9, 6, 11, 13, 7, 12

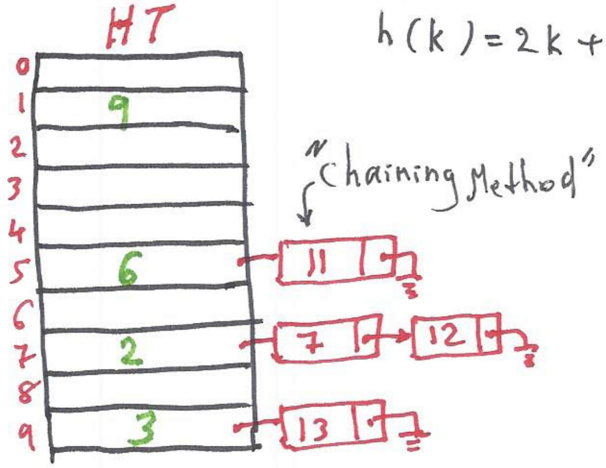
Hash func. → $H(k) = 2k+3$, $\boxed{m=10}$ $\underline{\text{Size of HT}}$

We use Division method (%) and open Hashing to store these values (KV).

We want to store these values in HT.

Division method: $h(k_i) = k_i \% m$

$h(k) = 2k+3 \longrightarrow \boxed{2k+3\%m}$



Key	Location u or index
3	$[2\times3+3]\%10 = 9$
2	$[2\times2+3]\%10 = 7$
9	$[2\times9+3]\%10 = 1$
6	$[2\times6+3]\%10 = 5$
11	$[2\times11+3]\%10 = \boxed{5}$ Collision
13	$[2\times13+3]\%10 = \boxed{9}$ Collision
7	$[7\times2+3]\%10 = \boxed{7}$ Collision
12	$[12\times2+3]\%10 = \boxed{7}$ Collision

Week 16, Sat: K.V. = 3, 2, 9, 6, 11, 13, 7, 12

ex #1b: We use "Open Addressing" with linear probing method (for Collision)

Insert k_i at first free location from $\boxed{(u+i)\%m}$, where $i = 0$ to $(m-1)$.

$\boxed{h(k) = 2k+3}$

m = 10 size of Hash table ↓ ↓

	HT
0	13
1	9
2	12
3	
4	
5	6
6	11
7	2
8	7
9	3

$h(k) = 2k+3$

Key	Location(u), index	probs
3	$(2\times3+3)\%10 = 9$	1
2	$(2\times2+3)\%10 = 7$	1
9	. . . . = 1	1
6	. . . . = 5	1
11	. . . . . . = 5 Collision	2
13	. . . . . . = 9 Collission	2
7	. . . . . . = 7 Collission	2
12	. . . . . . = 7 Collision	6

Linear Probing Method

* $\boxed{(u+i)\%m}$, let's check for key = $\underline{11}$, $(i=0 \text{ to } \overset{9}{(m-1)})$

$\{11 = (5+\underset{i=0}{0})\%10 \xrightarrow{=5} 11 = (5+\underset{i=1}{1})\%10 = 6$ ✓ (We send 11 to index 6) → probs = 2

$\{13 = (9+0)\%10 = 9 \rightarrow (9+1)\%10 = \underline{0}$ → (We send 13 to index 0) → probs = 2

$\{\ 7 = (7+0)\%10 = 7 \rightarrow (7+1)\%10 = 8$ → (We send 7 to index 8) → probs = 2

$12 = (7+\underset{i=0}{0})\%10 = 7 \rightarrow (7+\underset{i=1}{1})\%10 = 8 \rightarrow (7+\underset{i=2}{2})\%10 = 9 \rightarrow$

$(7+\underset{i=3}{3})\%10 = 0 \rightarrow (7+\underset{i=4}{4})\%10 = 1 \rightarrow (7+\underset{i=5}{5})\%10 = \underline{2}$

(We send 12 to index 2)

We tried 6 times to find a free location (probs = 6)

What is the order of elements (K.V.) in the HT?

13, 9, 12, –, –, 6, 11, 2, 7, 3

ex#1, C: Use "open addressing" with "Quadratic probing" to store the following key values and using division method (%). → $H(k_i) = k_i \% m$, m=10 size of HT

$k_i = 3, 2, 9, 6, 11, 13, 7, 12$

For Collision, insert k_i at first free location from $(u+i^2)\%m$.

Hash func. is $H(k) = 2k+3$

"Continue on next page"

↓ ↓

HT

Index	HT
0	13
1	9
2	
3	12
4	
5	6
6	11
7	2
8	7
9	3

Key	location (u), index		Probs.
3	$(2\times3+3)\%10=9$		1
2	$(2\times2+3)\%10=7$		1
9	.... $=1$		1
6	.... $=5$		1
11	.... $=\boxed{5}$	Collision	2
13	.... $=\boxed{9}$	"	2
7	.... $=\boxed{7}$	"	2
12	.... $=\boxed{7}$	"	5

$11=(\overset{u}{5}+0^2)\%10=5 \rightarrow (5+1^2)\%10=\underline{\underline{6}}$ (We send 11 to index 6)

$13=(\overset{u}{9}+0^2)\%10=9 \rightarrow (9+1^2)\%10=\underline{\underline{0}}$ (" " 13 " " 0)

$7=(7+0^2)\%10=7 \rightarrow (7+1^2)\%10=\underline{\underline{8}}$ (" " 7 " " 8)

$12=(7+0^2)\%10=7 \rightarrow (7+1^2)\%10=8 \rightarrow (7+2^2)\%10=1 \rightarrow$

$(7+3^2)\%10=6 \rightarrow (7+4^2)\%10=3 \rightarrow$ (We send 12 to index $\underline{\underline{3}}$)

Order of elements (KV): 13, 9, –, 12, –, 6, 11, 2, 7, 3

ex#1, d: Use Division method and double Hashing for Collision, $h_1(K)=2k+3$, $h_2(k)=3k+1$ technique to insert Key Values, $m=10$.

$k_i = 3, 2, 9, 6, 11, 13, 7, 12$

We use $h_2(k)$ in Case of Collision. When Collision happens, insert k_i at first free space from $(u+v*i)\%10$. $h_1(k)=2k+3$, $h_2(k)=3k+1$

H.T.

	H.T.
0	
1	9
2	
3	11
4	12
5	6
6	
7	2
8	
9	3

$(u+v*i)\%10$

$11=(5+4*0)\%10=5$

$(5+4\times1)\%10=9$

$(5+4\times2)\%10=\boxed{\underline{3}}$

$13=(9+0\times0)\%10=9$

$(9+0\times1)\%10=9$

$\vdots \qquad \vdots \quad =9$

(We $\underline{\text{cannot}}$ insert $\underline{\underline{13}}$ in H.T.)

Key	Location (u)	Location (V)	Probs
3	$(2\times3+3)\%10=9$	—	1
2	.... $=7$	—	1
9	.... $=1$	—	1
6	.... $=5$	—	1
11	.... $=\boxed{5}$	$(3\times11+1)\%10=\boxed{4}$	3
13	.... $=\boxed{9}$	$(3\times13+1)\%10=\boxed{0}$	x
7	.... $=\boxed{7}$	$(3\times7+1)\%10=\boxed{2}$	x
12	.... $=\boxed{7}$	$(3\times12+1)\%10=\boxed{7}$	2

$7=(7+2\times0)\%10=7 \rightarrow (7+2\times1)\%10=9 \rightarrow (7+2\times2)\%10=1 \rightarrow (7+2\times3)\%10=3$

$(7+2\times4)\%10=5 \rightarrow (7+2\times5)\%10=7 \rightarrow (7+2\times6)\%10=9 \rightarrow (7+2\times7)\%10=1$

$(7+2\times8)\%10=3 \rightarrow (7+2\times9)\%10=5 \rightarrow$ We $\underline{\text{cannot}}$ insert $\underline{\underline{7}}$.

$12=(\overset{u}{7}+\overset{v}{7}\times\overset{i}{0})\%10=7 \rightarrow (\overset{u}{7}+\overset{v}{7}\times\overset{i}{1})\%10=\boxed{4}$ (We send 12 to index 4).

Order of elements (KV): _ , 9, _ , 11, 12, 6, _ , 2, _ , 3

We failed for key values 13 and 7, so it is better to try another method (like using different $h_2(k)$).

- **Folding Method:** We divide the big number (key) into equal size pieces, we may not have the last piece of equal size. We add these pieces together to get the result. For example, if the key is a phone number 436-555-4601, we divide these digits into group of two. We get 43, 65, 55, 46, 01, now we add them up.

$$43+65+55+46+01=210,$$ if m=11 (HT size), we get $210\%11=1$.

We put this phone number in HT with index=1. Some folding methods reverse every other piece before adding.

In this example we get: 43, 56, 55, 64, 01, the total is 219 then $219\%11=10$, so we store the phone # in the index 10 of HT array.

Mid-Square Method: First we square each item and then we extract some portion of resulting digits. For example if value 44 → $44^2=1\boxed{93}6$.

We extract 93 then we do $93\%11=5$

Numbers	Remainder (index for HT)
$54 \rightarrow 54^2=2\boxed{91}6$	$91\%11=3$
$26 \rightarrow 26^2=6\boxed{7}6$	$7\%11=7$
$93 \rightarrow 93^2=8\boxed{64}9$	$64\%11=9$
$17 \rightarrow 17^2=2\boxed{8}9$	$8\%11=8$
$77 \rightarrow 77^2=5\boxed{92}9$	$92\%11=4$
$31 \rightarrow 31^2=9\boxed{6}1$	$6\%11=6$

0	1	2	3	4	5	6	7	8	9	10
			54	77		31	26	17	93	

Load factor

$$\lambda = \frac{\text{\# of values in HT}}{\text{table size}} = \frac{6}{11} = 0.5454 \text{ or } 54.54\%$$

$$S(\lambda) = \frac{1}{2}\left(1+\frac{1}{1-\lambda}\right) \rightarrow$$ Avg # of probs for successful search.

$$U(\lambda) = \frac{1}{2}\left(1+\frac{1}{(1-\lambda)^2}\right)$$ " " " " " unsuccessful " .

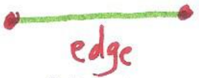
Graph Theory: A graph is a non-empty set of vertices and set of edges.

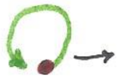
$$G = (V, E)$$

V: vertex set (also called nodes or points)

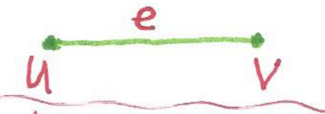
E: edge set (" " links, lines, arcs)

Each edge has either one or two vertices associated with it, called **endpoints**.
An edge is said to connect its endpoints.



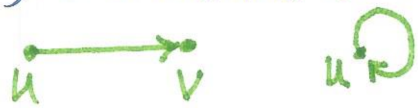

→ This is a loop.

- Vertices are said to be adjacent (or neighbors) if the vertices are endpoints of an edge.

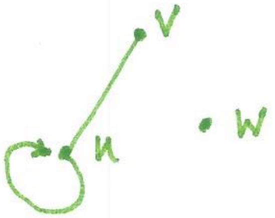


Edge $\underline{\underline{e}}$ is <u>incident</u> with $\underline{\underline{u}}$, $\underline{\underline{v}}$.

A directed graph (**Digraph**) Consists of set of vertices and directed edges.
The directed edge is associated with $\{u, v\}$ is said to start at $\underline{\underline{u}}$ and end at $\underline{\underline{v}}$.



The **degree** of a vertex in an undirected graph is the # of edges incident with it. A **loop** Contributes twice to the degree, we denote as **deg(V)**.



deg(u) = 3

deg(v) = 1 (**Pendant Vertex**)

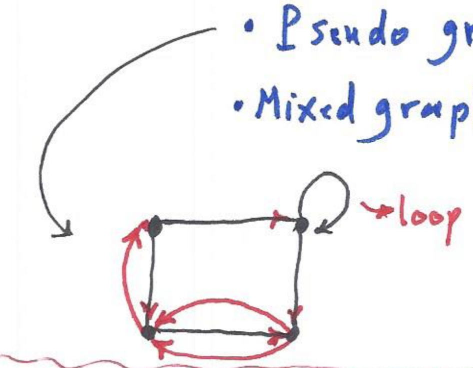
deg(w) = 0 (**Isolated Vertex**)

Graph Theory Applications:

- Electrical Eng.: To design circuits
- Computer Science: Dijkstra's Alg. (to find the shortest path)
  - prim's Alg (Min. Spanning Tree)
  - Kruskal's Alg.
- Computer Networking
- Science: The molecular structure and chemical structure of a substance, the DNA structure of an organism, etc. are represented by graphs.
- Linguistics: The "parsing tree" of a language and grammer of a language
- General: Business (Optimization), it Can be Min or Max.

Graph Types : • Simple: no loops, no multiple or parallel edges

- Multi-graph: " ", yes multiple " " "
- Pseudo graph: yes loops, yes multiple " " ".
- Mixed graph: has directed and undirected edges.



Dijkstra's Algorithm: We use it to find the shortest path from a single source vertex to all vertices in graph.

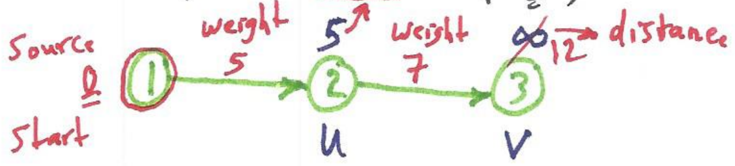
1- $\overbrace{\text{Shortest path tree}}^{S.p.t}$ set: empty

2- Assign distance values as ∞. For source (start) vertex assign 0.

3- a) pick a vertex u which is not in the s.p.t. set.

b) put this vertex u in s.p.t. set

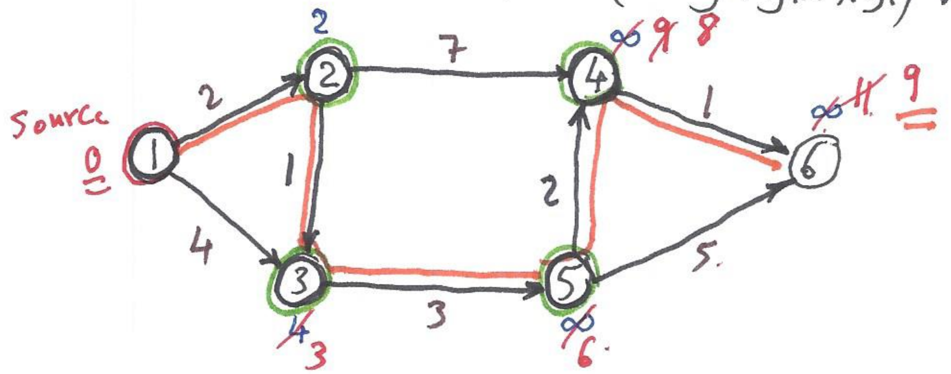
c) update all distance values of all adjacent vertices of u. It is called relaxation. For every adjacent vertex V, if sum of distance value of u (from source) and weight of edge uv is less than the value of V, then update distance of V,



There is no directed path from vertex 1 to vertex 3, we use ∞.

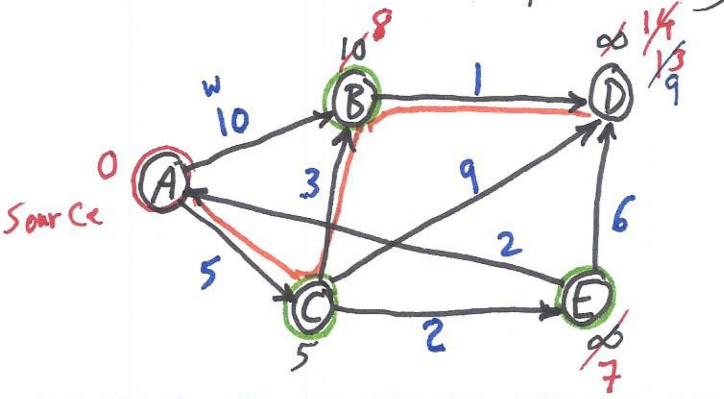
$$5+7=12<\infty \longrightarrow \text{relaxation}: \begin{cases} \text{if } \overset{\nearrow \text{distance}}{d}[u]+\overset{\nearrow \text{Cost}}{C}(u,v)<d[V] \\ \text{then } d[V]=d[u]+C(u,v) \end{cases}$$

ex#1, Find the shortest path (using Dijk. Alg.):



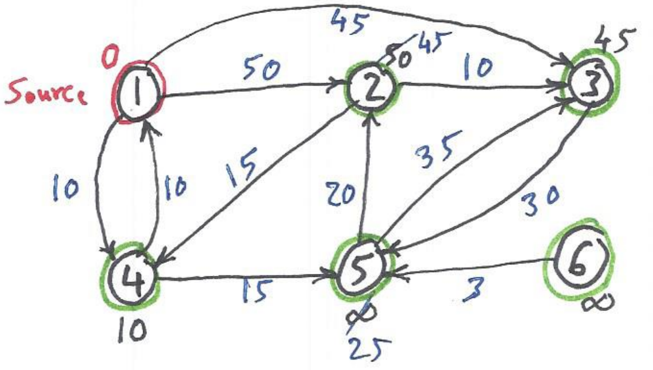
V	d[V]
1	0
2	2
3	3
4	8
5	6
6	9

ex#2, Find the shortest path: using table, start with A



Selected Vertex	A	B	C	D	E
A	0	∞	∞	∞	∞
C		10	5	∞	∞
E		8		14	7
B		8		13	
D				9	

ex#3, Find the shortest path from Source 1 using table:



Selected Vertex	1	2	3	4	5	6
1	0	∞	∞	∞	∞	∞
4		50	45	10	∞	∞
5		50	45		25	∞
2		45	45			∞
3			45			∞
6						∞

Degree of Vertex: 1- undirected graph   2- Directed graph

In a simple graph with n # of Vertices, the degree of any vertices is:

$$\deg(v) \leq n-1 \quad \underset{\text{every or all}}{\forall}\, v \in G \rightarrow \text{belongs}$$

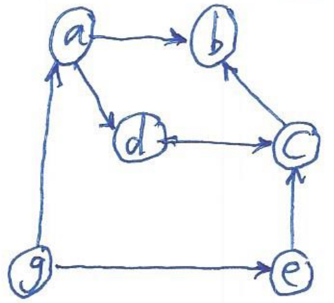
For directed graph: 1- indegree   2- outdegree

- Indegree of a graph: Indegree of Vertex V is the # of edges which are Coming into vertex V.

$$\deg \underset{\text{negative}}{-} (V)$$

- Outdegree of a graph: Out degree of Vertex V is the # of edges which are going out from the vertex V.

$$\deg + (V)$$



Vertex	Indegree	outdegree
a	1	2
b	2	0
c	2	1
d	1	1
e	1	1
g	0	2

- Representation of Graph in Computer:

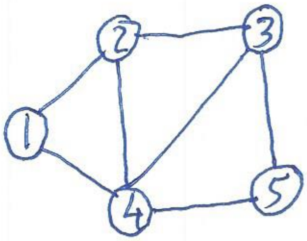
1- Adjacency Matrix: $A_{n \times n} \Rightarrow \underline{\underline{n}}$ # of vertices

$a_{ij} = 1$, if $\underline{\underline{i}}$ and $\underline{\underline{j}}$ are adjacent.

Otherwise $a_{ij} = 0$

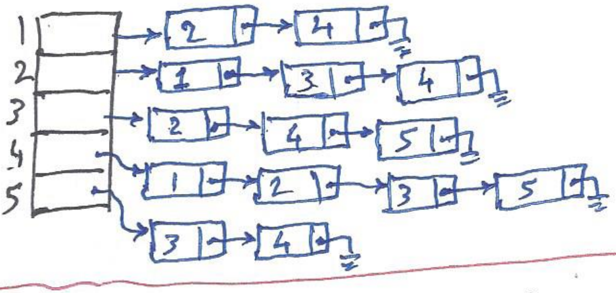
2- Adjacency list

ex#1, write adjacency matrix for the following graph:



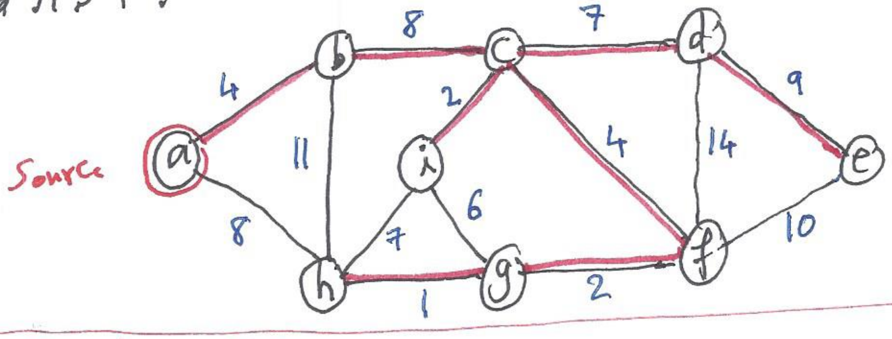
$$\longrightarrow \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 1 & 1 & 0 \\ 3 & 0 & 1 & 0 & 1 & 1 \\ 4 & 1 & 1 & 1 & 0 & 1 \\ 5 & 0 & 0 & 1 & 1 & 0 \end{array} \longrightarrow$$ It is a 2D- Array.

Adjacency List:



prim's Algorithm: Minimum Spanning Tree (MST), Min. Cost to visit all vertices. We select least cost path at each vertex which is connecting other vertices. We cannot have a cycle.

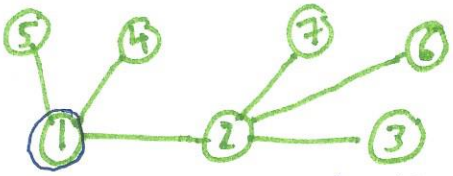
Example: Find MST:



- Breadth First Search: BFS

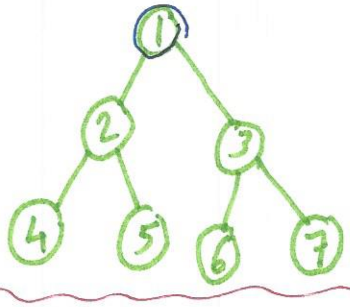
1- Visit a vertex and its neighbors and do the successively. We can start from any vertex unless is given.

2- Exploration of vertex



BFS: 1 2 4 5 3 6 7

- Depth First Search: DFS, We select a path based on numerical incrementor dictionary order.



DFS: 1 2 4 5 3 6 7

BFS: 1 2 3 4 5 6 7

Final Exam is next Sat. 6-1-24, 10AM-12PM

Topics: 1- Exceptions 2- Recursion 3- Binary Tree, BST, Heap 4- Hashing & Graphs