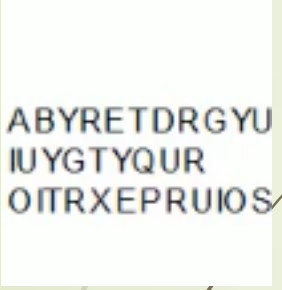
Data Structure and Algorithms

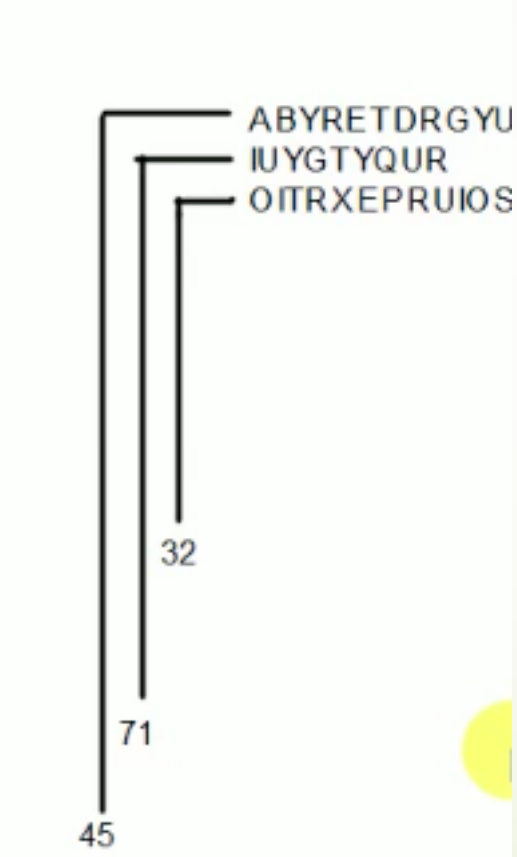
Session-30

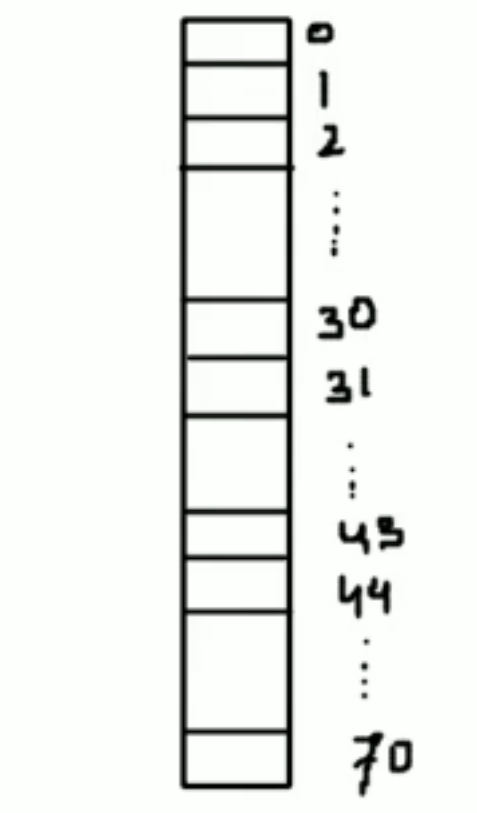
Dr. Subhra Rani Patra
SCOPE, VIT Chennai

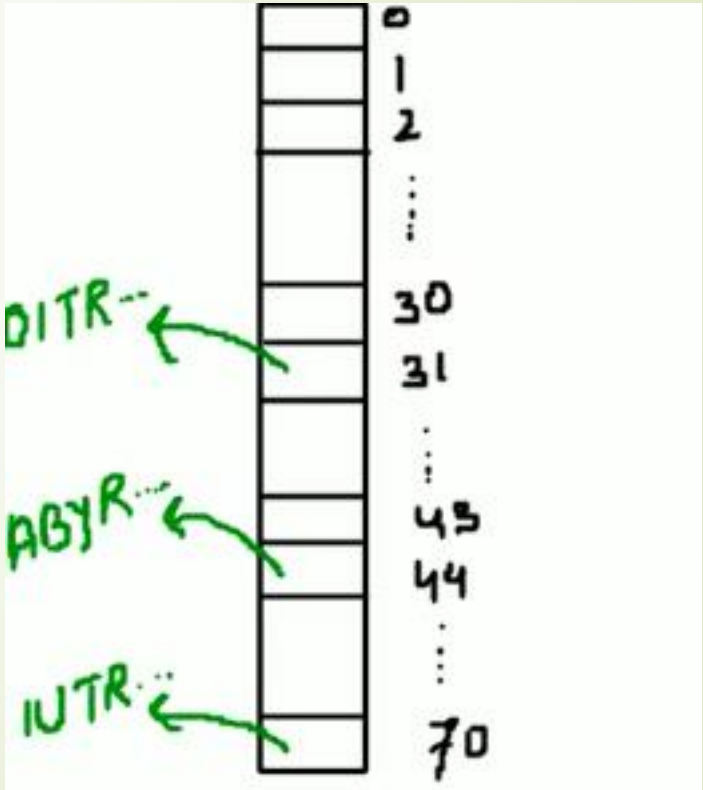
What is Hashing:

- Hashing is a method of sorting and indexing data. The idea behind hashing is to allow large amounts of data to be indexed using keys commonly created by formulas.









Why we need Hashing ?

✓ Time efficient:

Data Structures	Time Complexity for Search operation
Array	O(log n)
Linked List	O(n)
Tree	O(log n)
Hashing	O(1) / O(n)

Linear Search – O(n)

	0	1	2	3	4	5	6	7	8	9
A	8	5	12	6	15	9	4	3	7	10

Binary Search – O(log n)

	0	1	2	3	4	5	6	7	8	9
A	3	4	5	6	7	8	9	10	12	15

Hashing

Keys — 8, 3, 13, 6, 4, 10

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A				3	4		6		8		10			13	

Some Terminologies:

- **Hash Function:** A hash function is any function that can be used to map data of arbitrary size to data of fixed size.
- **Key:** Input data given by user
- **Hash Value:** The values returned by a hash function are called hash values, hash codes, digests, or simply hashes.
- **Hash table:** It is a data structure which implements an associative array abstract data type, a structure that can map keys to values.
- **Collision:** A collision occurs when two different key to a hash function produce the same output called hash value.

Sample good Hash Function:

- Simple Mod Function (for integer Inputs):
- ASCII Functions (for String Inputs):
- Etc…

Basic01.java

```java
/**
package basics;
public class Basic01 {

    public static void main(String[] args)
    {
        System.out.println(smod(500));

    }

    static int  smod(int M) {
          return M % 16;
        }
```

Javadoc | Console | Progr

```
<terminated> Basic01 [Java Application]
4
```

16 | 500 | 31
496
4

16 | 700 | 43
688
12

Basic01.java

```java
/**
package basics;
public class Basic01 {

    public static void main(String[] args)
    {
        System.out.println(sascii("ABCDEFGH",16));

    }

    static int  sascii(String x, int M) {
          char ch[];
          ch = x.toCharArray();
          int i, sum;
          for (sum=0, i=0; i < x.length(); i++)
            sum += ch[i];
          return sum % M;
        }

}
```

Javadoc | Console | Progress | Synchronize

```
<terminated> Basic01 [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe
4
```

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~

$$65 + 66 + 67 + 67 + 68 + 69 + 70 + 71 + 72 = \mathbf{548}$$

Basic01.java

```java
/**
package basics;
public class Basic01 {

    public static void main(String[] args)
    {
        System.out.println(sascii("ABCDEFGH",16));

    }

    static int  sascii(String x, int M) {
          char ch[];
          ch = x.toCharArray();
          int i, sum;
          for (sum=0, i=0; i < x.length(); i++)
            sum += ch[i];
          return sum % M;
        }

}
```

Javadoc | Console | Progress | Synchronize

```
<terminated> Basic01 [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe
4
```

16 | 548 | 34
544
4

Characteristics of good Hash Function:

- It distributes hash values uniformly across the hash table.
- The hash function uses all the input data.

Basic01.java

```java
/**
package basics;
public class Basic01 {

    public static void main(String[] args)
    {
        System.out.println(sascii("ABCDEFGH",16));

    }

    static int  sascii(String x, int M) {
          char ch[];
          ch = x.toCharArray();
          int i, sum;
          for (sum=0, i=0; i < x.length(); i++)
            sum += ch[i];
          return sum % M;
        }

}
```

ABCDEFGH
KLMNOPQR
PQRSTUVW

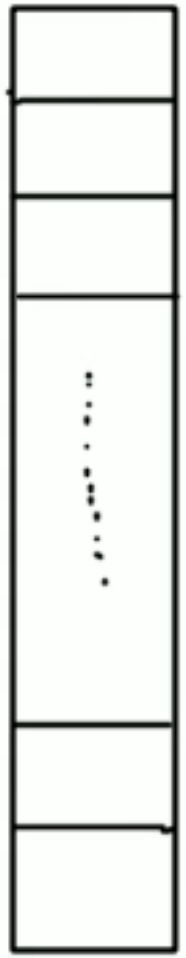
444

Basic01.java

```java
/**
package basics;
public class Basic01 {

    public static void main(String[] args)
    {
        System.out.println(sascii("ABCDEFGH",16));

    }

    static int  sascii(String x, int M) {
          char ch[];
          ch = x.toCharArray();
          int i, sum;
          for (sum=0, i=0; i < x.length(); i++)
            sum += ch[i];
          return sum % M;
        }

}
```

```
ABCDEFGHIJKLMNOP
ABCDEFGHRRRRRRRR
ABCDEFGHPQWERTY
```

ABCDEFG
PQRSTUV
MNOPQRS

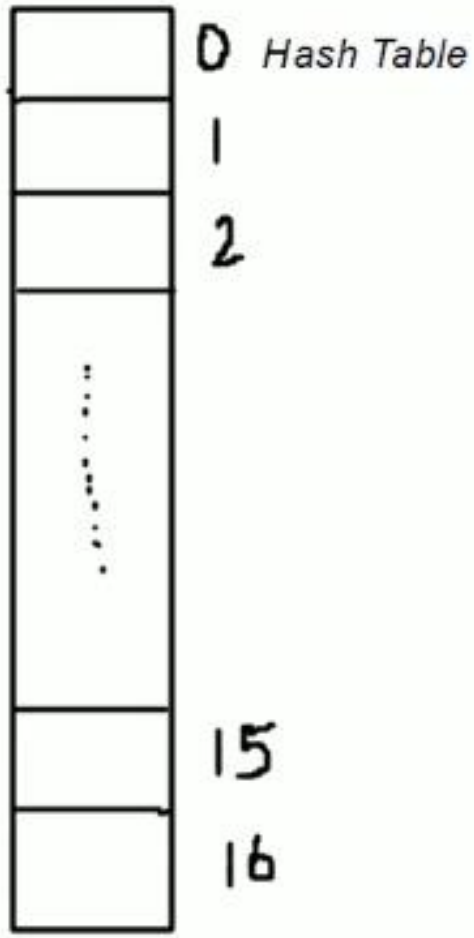


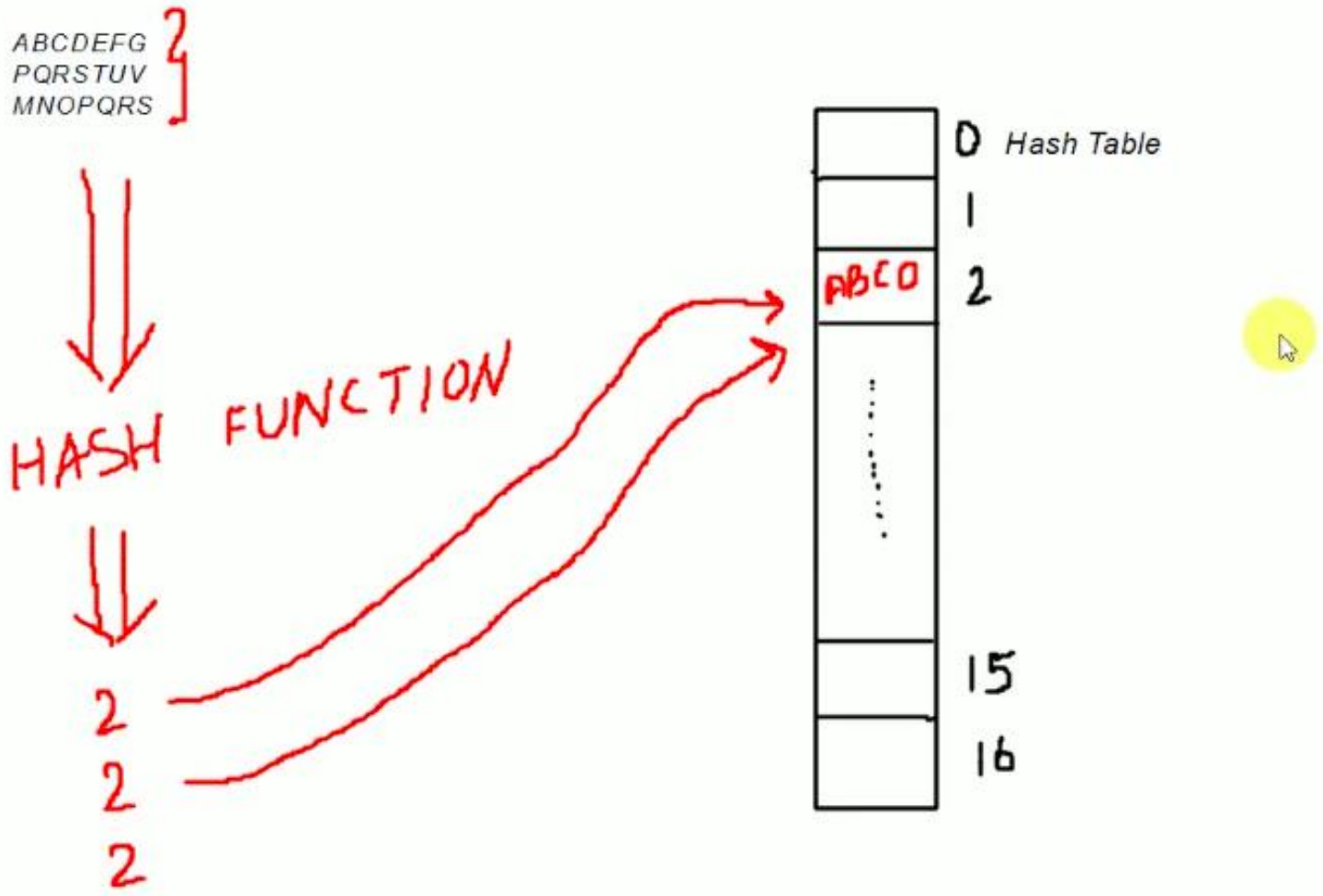
Hash Table

ABCDEFG
PQRSTUV
MNOPQRS

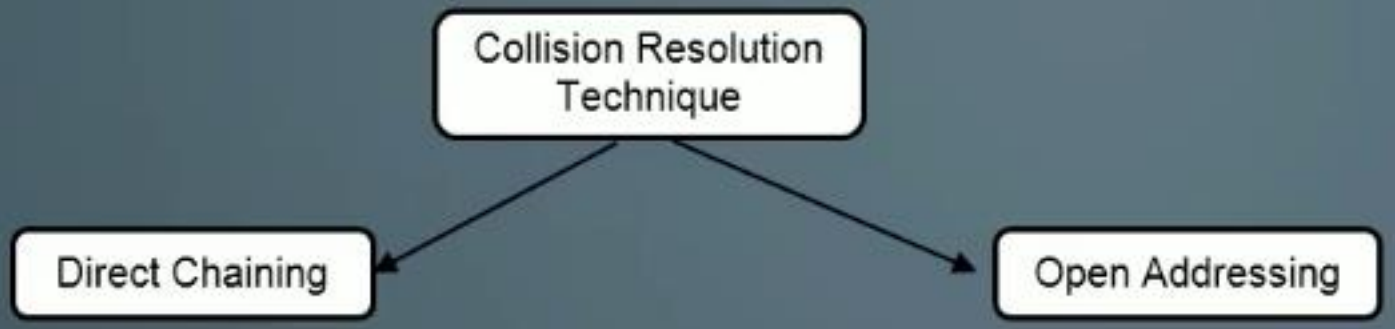
HASH FUNCTION

2
2
2

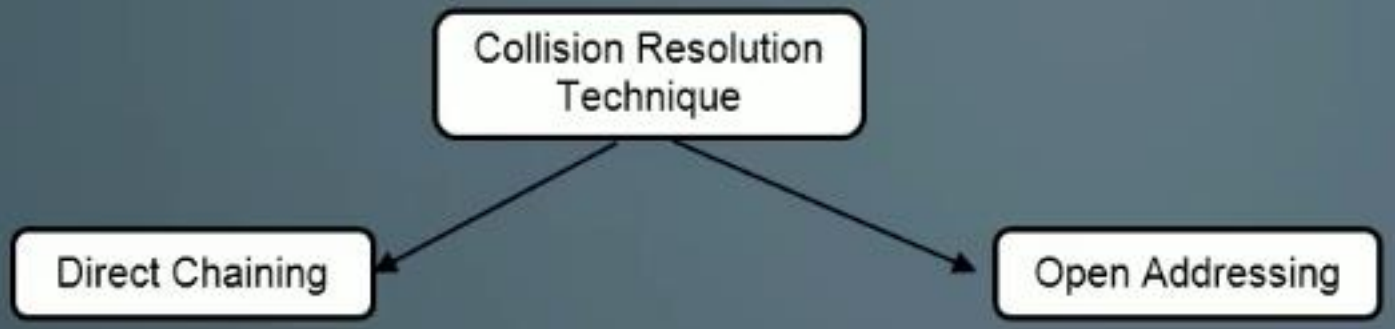



ABCDEFG
PQRSTUV
MNOPQRS
HASH FUNCTION
2
2
2
0
Hash Table
1
ABCD
2
15
16

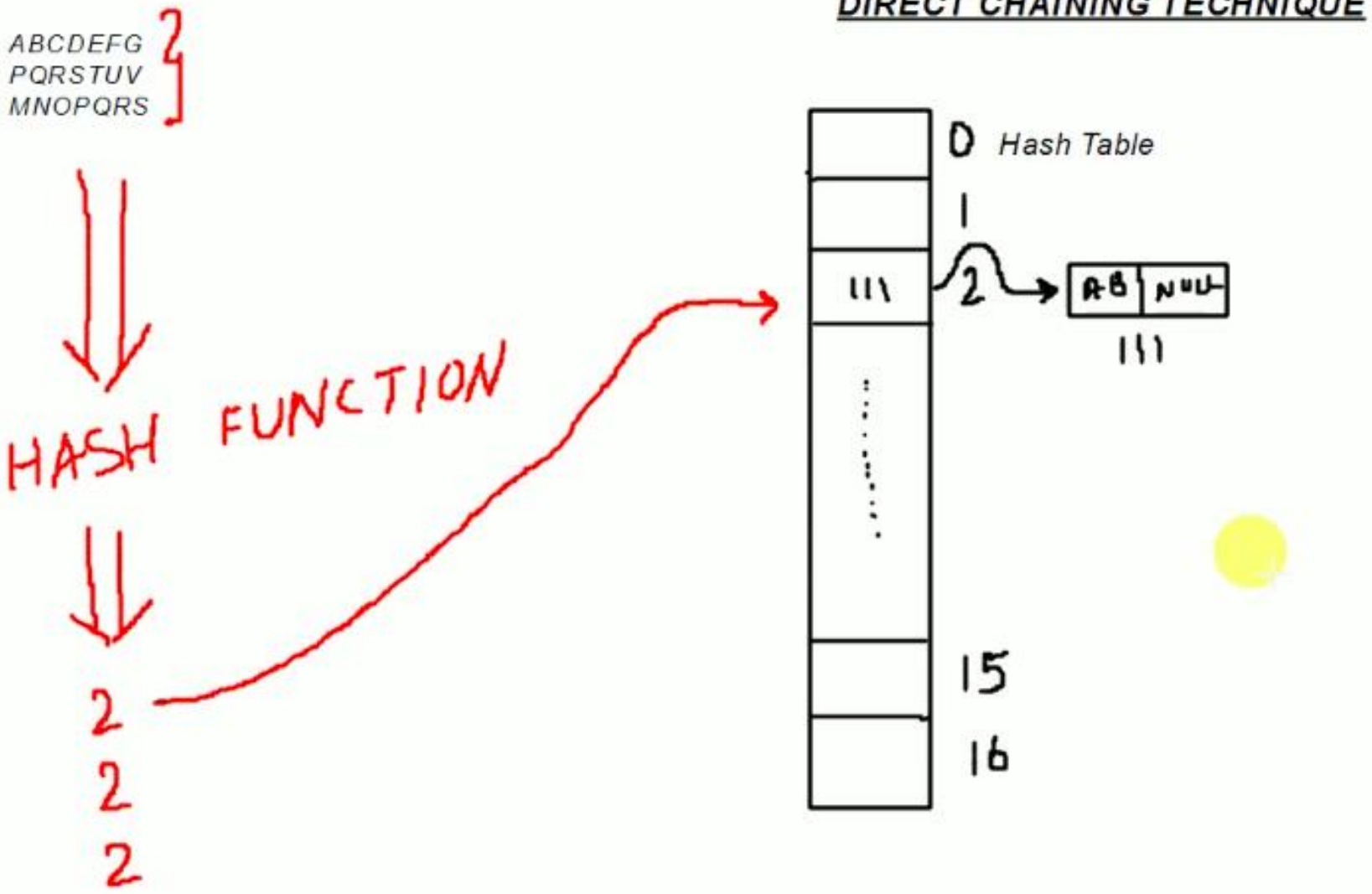
Collision Resolution Techniques:



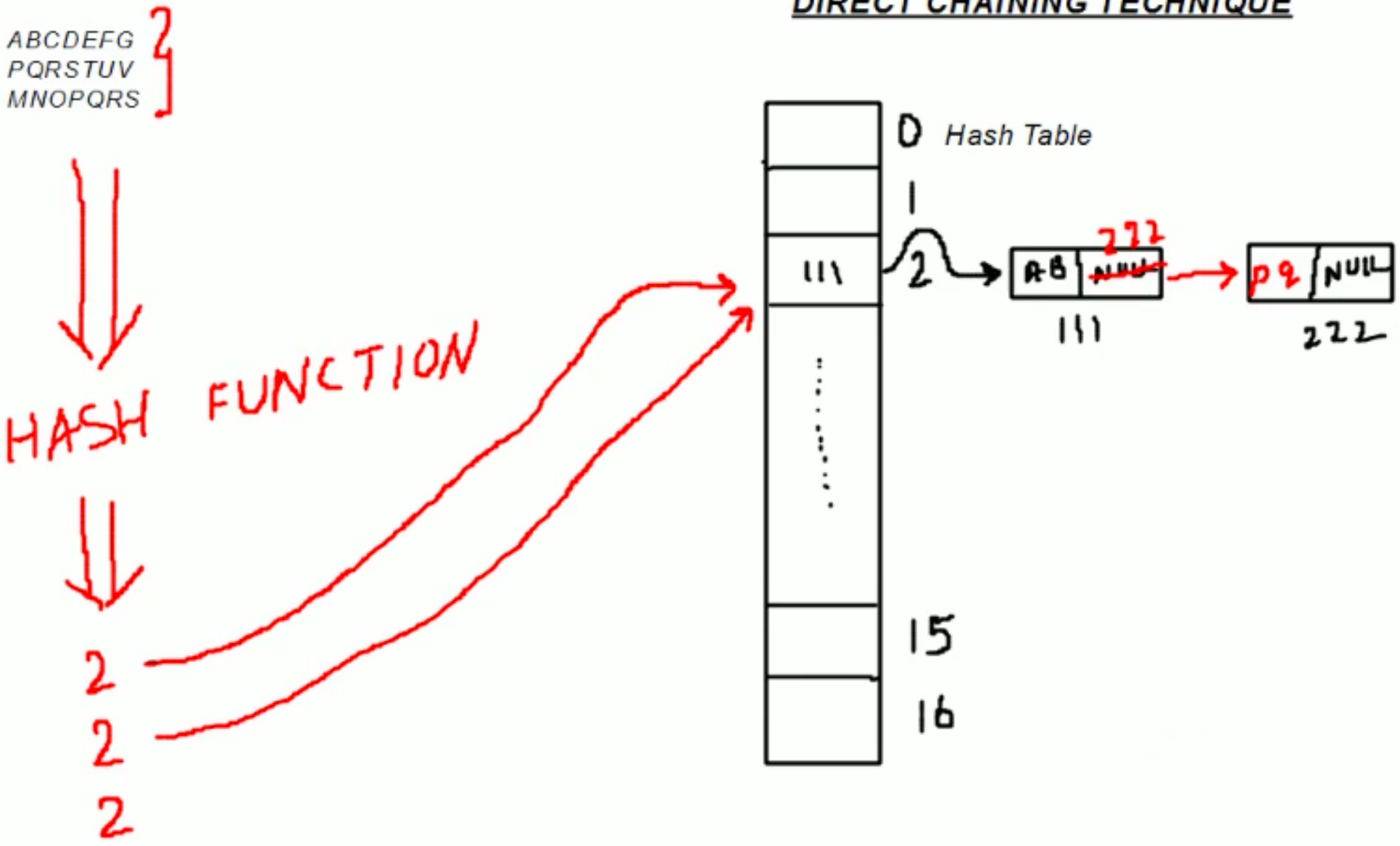
Collision Resolution Techniques:



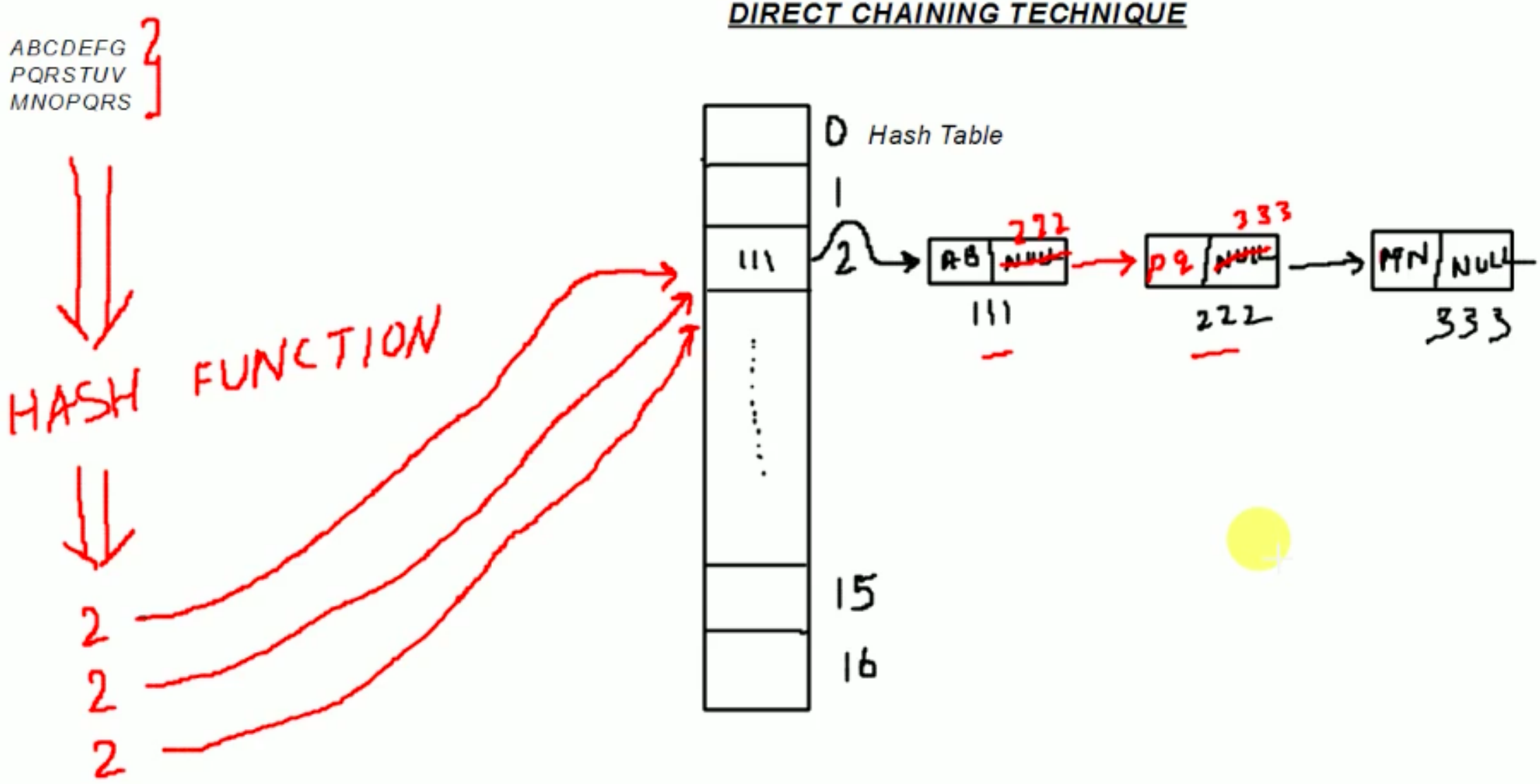
DIRECT CHAINING TECHNIQUE



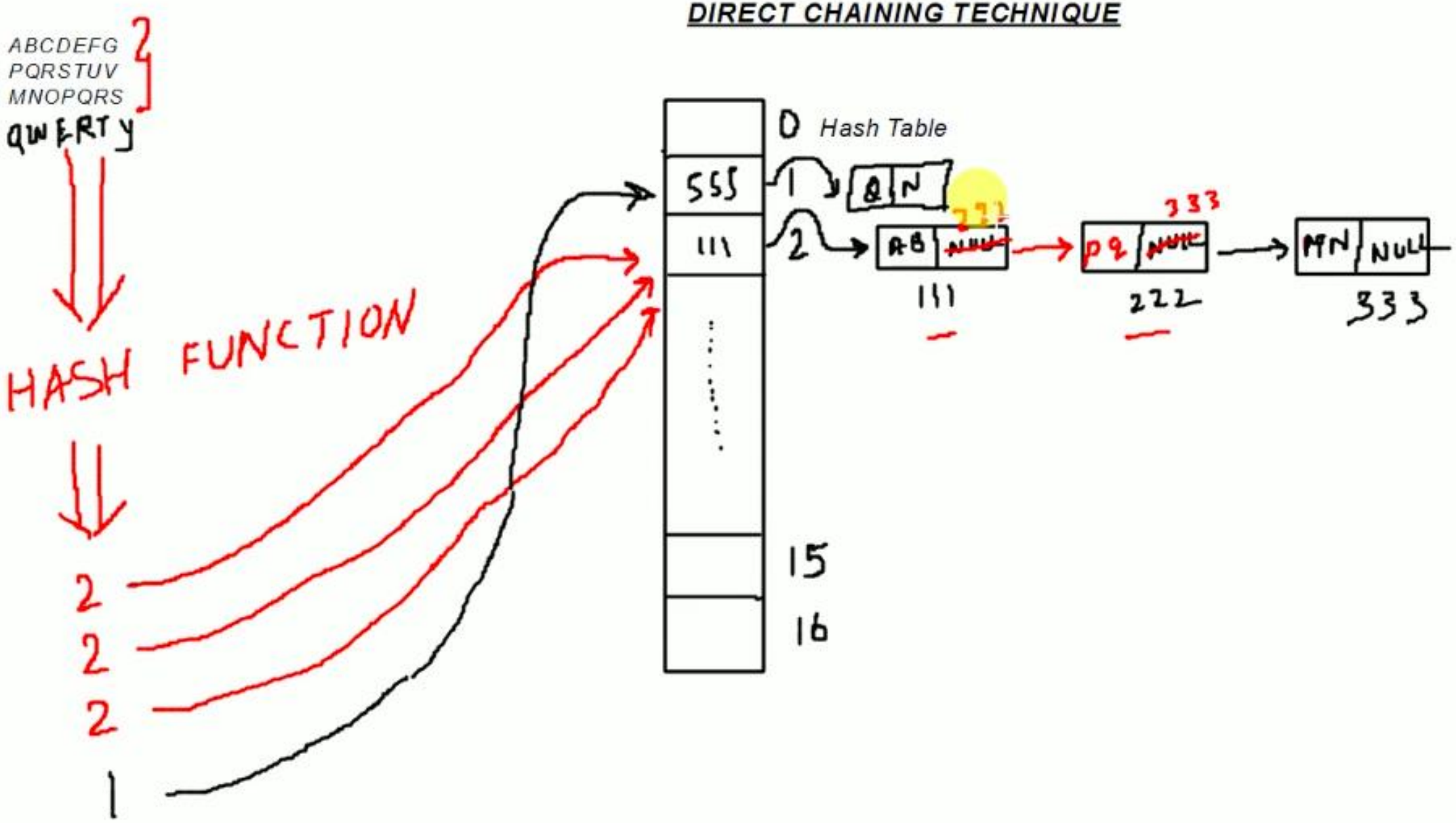
DIRECT CHAINING TECHNIQUE



DIRECT CHAINING TECHNIQUE



DIRECT CHAINING TECHNIQUE




Thank you