NAME :	Maryada Lodha
EXPERIMENT NO :	10
DATE :	07-12-2022

AIM :	Implementation of Hashing by Linear Probing

ALGORITHM :

int main()

1] Read the tableSize from the user

2] Initialize the array hashTable of size tableSize to 0

3] Set counter to 0 and repeat steps 4,5,6 and 7 while counter < tableSize

4] Read the Key to be inserted from the user

5] Calculate index as h = Key % tableSize

6] If hashTable[h] != 0

i] Print Collision at hashTable[h]

ii] Set i = 1 and repeat steps [iii] and [iv] while i < tableSize + 1

iii] Calculate h1 = (h + i) % tableSize

iv] If hashTable[h1] = = 0

Set hashTable[h1] = Key and break

Else

Print Collision at hashTable[h1], increment i and go to step [iii]

[End of If]

Else

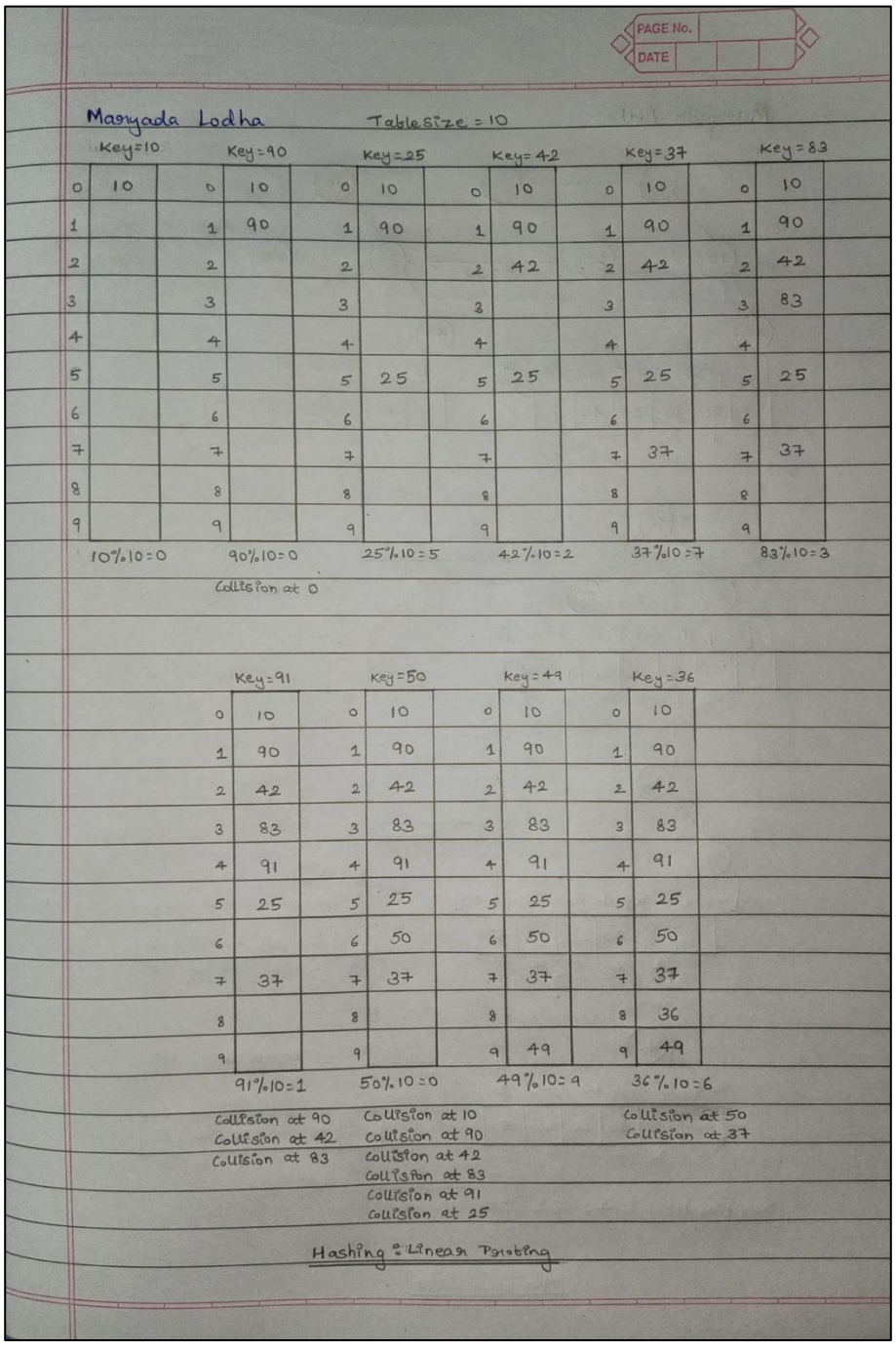
Set hashTable[h] = Key

[End of If]

7] Increment counter

8] End

	void display(int hashTable[],int tableSize) 1] Set n = 0 and repeat step 2 while n < tableSize 2] Print hashTable[n] and increment n 3] End

PROBLEM SOLVING :	

CODE :	#include <stdio.h>
	void display(int hashTable[],int tableSize);
	int main()
	{
	int tableSize,key,h,h1,n,i,counter=0;
	printf("\nEnter the Size of the Table : ");
	scanf("%d",&tableSize);
	int hashTable[tableSize];
	
	for(n=0;n<tableSize;n++)
	{
	hashTable[n]=0;
	}
	
	while(counter<tableSize)
	{
	printf("\n\nEnter Key to Insert : ");
	scanf("%d",&key);
	h=key%tableSize;
	if(hashTable[h]!=0)
	{
	printf("\nCollision at %d",hashTable[h]);
	for(i=1;i<tableSize+1;i++)
	{

```
            h1=(h+i)%tableSize;
            if(hashTable[h1]==0)
            {
               hashTable[h1]=key;
               break;
            }
            else
            {
               printf("\nCollision at %d",hashTable[h1]);
            }
         }
      }
      else
      {
         hashTable[h]=key;
      }
      display(hashTable,tableSize);
      counter++;
   }
   return 0;
}

void display(int hashTable[],int tableSize)
{
   int n;
```

```
    printf("\n\nHash Table : ");
    for(n=0;n<tableSize;n++)
    {
        printf("%d   ",hashTable[n]);
    }
}
```

OUTPUT :

```
Enter the Size of the Table : 10

Enter Key to Insert : 10

Hash Table : 10   0   0   0   0   0   0   0   0   0
Enter Key to Insert : 90
Collision at 10
Hash Table : 10   90   0   0   0   0   0   0   0   0
Enter Key to Insert : 25

Hash Table : 10   90   0   0   0   25   0   0   0   0
Enter Key to Insert : 42

Hash Table : 10   90   42   0   0   25   0   0   0   0
Enter Key to Insert : 37

Hash Table : 10   90   42   0   0   25   0   37   0   0
Enter Key to Insert : 83

Hash Table : 10   90   42   83   0   25   0   37   0   0
```

```
Enter Key to Insert : 91

Collision at 90
Collision at 42
Collision at 83

Hash Table : 10   90   42   83   91   25   0   37   0   0

Enter Key to Insert : 50

Collision at 10
Collision at 90
Collision at 42
Collision at 83
Collision at 91
Collision at 25

Hash Table : 10   90   42   83   91   25   50   37   0   0

Enter Key to Insert : 49


Hash Table : 10   90   42   83   91   25   50   37   0   49

Enter Key to Insert : 36

Collision at 50
Collision at 37

Hash Table : 10   90   42   83   91   25   50   37   36   49
```