TUTORIAL-1

1) d) All 1, 2 & 3

2) Array p: 5 7 11 17 25
Array q: 25 17 11 7 5

3)
```
#include <iostream>
using namespace std;
int main()
{
int *a;
a = new int [5];
for(int i=0;i<5;i++)
{
   cin>>a[i];
}

delete []a;

   return 0;
}
```

4)
```
#include <iostream>
using namespace std;
int main()
{
int *a;
a = new int [5];
for(int i=0;i<5;i++)
{
   cin>>a[i];
}
int *b,n=10;
b = new int [n];
for(int i=0;i<n/2;i++)
{
   b[i]=a[i];
}
for(int i=n/2;i<n;i++)
{
   b[i]=2*a[i-n/2];
}
for(int i=0;i<n;i++)
{
   cout<<b[i]<<" ";
}
delete []a;
delete []b;

   return 0;
```

```
}
5)
#include <iostream>
using namespace std;
struct Node {
  int data;
  struct Node* next;
}*head=NULL;

void insertNode( int new_data) {
    struct Node* new_node = new Node();

  new_node->data = new_data;
  new_node->next = head;
  head = new_node;
}
void deleteItem( int key) {
  struct Node *temp = head, *prev;

  if (temp != NULL && temp->data == key) {
  head = temp->next;
  free(temp);
  return;
  }
  while (temp != NULL && temp->data != key) {
  prev = temp;
  temp = temp->next;
  }
  if (temp == NULL) return;
  prev->next = temp->next;

  free(temp);
}

void display(struct Node*p) {
  while (p!= NULL) {
  cout <<p->data << " ";
  p =p->next;
  }
}

int main() {
  insertNode(1);
  insertNode(2);
  insertNode(3);
   cout << "Linked list: ";
  display(head);
  cout << "\nAfter deleting an element: ";
  deleteItem( 3);
  display(head);
}
```

TUTORIAL-2

1)
1st Ary before swapping are : 11 21 31 41 51 61
2nd Ary before swapping are: 71 81 91 10 11 12
1st Ary after swapping are: 71 81 91 10 11 12
2nd Ary after swapping are: 11 21 31 41 51 61

2)

```
#include <iostream>
#include <list>
using namespace std;

int main()
{
   list<int> list1;
   list1.push_back(3);
   list1.push_back(6);
   list1.push_back(2);
   list1.push_back(9);
   list1.push_back(1);
   list<int>::iterator i ;
   cout << "The list after inserting:";
   for (i = list1.begin();i != list1.end();i++)
      cout << *i << " ";
}
```

3)
Initial List: 1 2 3 4
List after first resize: 1 2
List after second resize: 1 2 0 0
List after third resize: 1 2 0 0 50

4)

```
#include <iostream>
#include <forward_list>
using namespace std;

int main()
{
   forward_list<int> list1;
   list1.push_front(10);
   list1.push_front(34);
   list1.push_front(56);
   list1.push_front(78);
   list1.push_front(34);
   list1.push_front(78);
```

```
   list1.push_front(90);
   forward_list<int>::iterator i ;
   cout << "The list after inserting:";
   for (i = list1.begin();i != list1.end();i++)
      cout << *i << " ";
}
```

TUTORIAl-3

1)

```
#include <iostream>
#include <stack>
using namespace std;
int main() {
        stack<int> st;
        st.push(1);
        st.push(2);
        st.push(4);
        st.push(5);

   st.pop();
        st.pop();

        while (!st.empty()) {
                cout << ' ' << st.top();
                st.pop();
        }
}
```

2)

```
#include <iostream>
#include <queue>

using namespace std;

int main()
{
   int sum=0;
   queue<int> q;
   q.push(3);
   q.push(7);
   q.push(4);
   q.push(5);
   q.push(1);
   while (!q.empty()) {
      sum+= q.front();
      q.pop();
   }
   cout<<sum;

   return 0;
```

```
}
```
3)
```
// CPP program to illustrate
// Implementation of swap() function
#include <stack>
#include <iostream>
using namespace std;

int main()
{
        stack<int> stack1;
        stack<int> stack2;
        stack1.push(41);
        stack1.push(33);
        stack1.push(20);
        stack1.push(11);

        stack2.push(90);
        stack2.push(75);
        stack2.push(58);
        stack2.push(35);

        swap(stack1, stack2);
   cout<<"stack1: ";
        while (!stack1.empty()) {
                cout<<stack1.top()<<" ";
                stack1.pop();
        }
   cout<<"stack2: ";
        while (!stack2.empty()) {
                cout<<stack2.top()<<" ";
                stack2.pop();
        }
        return 0;
}
```
4)
Output:
2
1
5) Reverse the entire queue
6)
Output: 34 76

TUTORIAL-4

1)

a) if(number==0)

b) return(number + mystery(number - 1));

c) Any nonnegative integer.

d)YES

e)YES

f) No as there is no base case for number <0

2)

```
int multiply(intn,int m)
{
If(n==0)
Return 0;
If(n==1)
Return m;

Return(m+multiply(m,n-1));
}
```

3)

```
#include <bits/stdc++.h>
using namespace std;
int first_occ(int arr[], int low, int high, int x, int n)
{
	if (high >= low) {
		int mid = low + (high - low) / 2;
		if ((mid == 0 || x > arr[mid - 1]) && arr[mid] == x)
			return mid;
		else if (x > arr[mid])
			return first_occ(arr, (mid + 1), high, x, n);
		else
			return first_occ(arr, low, (mid - 1), x, n);
	}
	return -1;
}

int last_occ(int arr[], int low, int high, int x, int n)
{
	if (high >= low) {
		int mid = low + (high - low) / 2;
		if ((mid == n - 1 || x < arr[mid + 1]) && arr[mid] == x)
			return mid;
		else if (x < arr[mid])
			return last_occ(arr, low, (mid - 1), x, n);
		else
			return last_occ(arr, (mid + 1), high, x, n);
	}
	return -1;
}
```

```
int main()
{
	int arr[] = { 10, 20, 20, 30, 20, 30, 40, 70, 80, 80 };
	int n = sizeof(arr) / sizeof(int);

	int x = 20;
	cout<<"First Occurrence of 20:"<<first_occ(arr, 0, n - 1, x, n);
   cout<<"\nLast Occurrence of 20:"<<last_occ(arr, 0, n - 1, x, n);
	return 0;
}
```

TUTORIAL -5

1)

Keys having collison: 1989,4199,6171

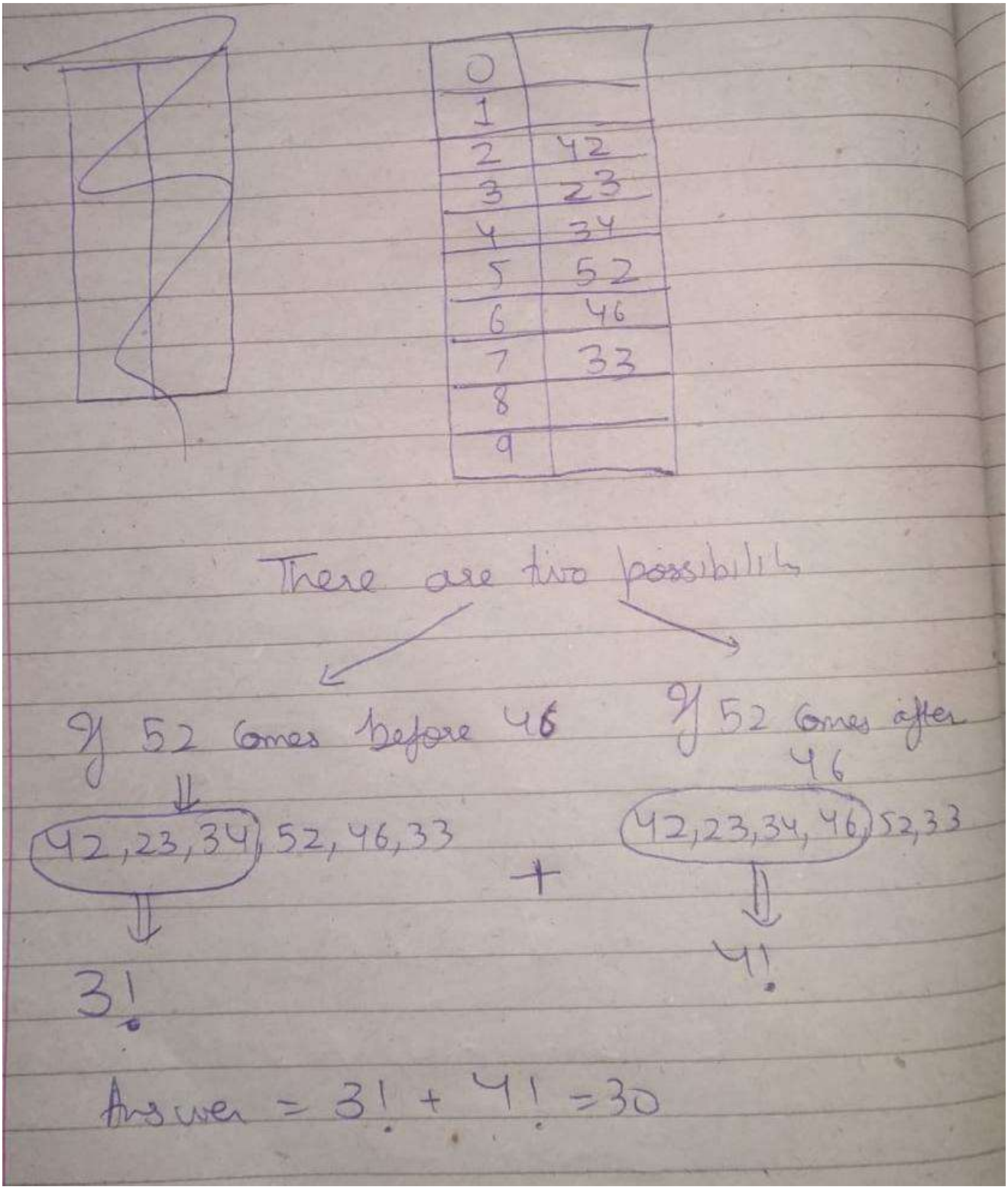
2)

0	
1	
2	12
3	13
4	2
5	3
6	23
7	5
8	18
9	15

3)

(C) 46, 34, 42, 23, 52, 33

4)



5)

$$\frac{97}{100} * \frac{97}{100} * \frac{97}{100} = 0.91273$$