NAME :	Tushar Srivastav
Univ. Rollno :	2014920
Section :	B
Class Rollno :	60
Subject :	Design And Analysis Of Algorithm
Subject code :	TCS 505

Ans 1-6 → Done in Assignment 1

Ans 7 =
```
int main()
{
    int n, key;
    bool flag = false;
     cin >> n;
     Vector <int> v(n);
       for (int i = 0; i<n; i++)
       { cin >> v[i];
       }
       cin >> key;

       map <int, int> mp;
       for (int i=0 ; i<n ; i++)
       {  int temp = key-v[i];
           if (mp.find (temp) == mp.end ())
             {  mp [v[i]] = i;
             }
             else
              {  cout << i << "  " << mp[x];
                  flag = true;
                  break;
              }
           }
           if (flag == false)
           {
               cout << "No such pair exist";
           }
           return 0;
       }
```

Ans 8 = QuickSort is the fastest general purpose sort. In most practical Situations, quicksort is the method of choice.
If stability is important and space is available, merge sort might be best.

Ans 9 = Insertion count for an array indicates how far (or close) the array is from being sorted. If the array is already sorted then the inversion count is 0 but if the array is sorted in the reverse order, the inversion count is the maximum.

Ans 10 = The best case for quicksort is when middle element is picked as a pivot.

The worst case for Quick Sort is when array is sorted in either increasing or decreasing order.

Ans 12 =
```
Void selection (int arr[], int n)
{ for (int i=0 ; i<n-1 ; i++)
    { int min = i;
      for (int j = i+1 ; j<n ; j++)
        { if (arr[min] > arr[j])
            { min = j;
            }
        }
        int key = arr[min];
        while (min > i)
        { arr[min] = arr[min-1];
            min--;
        }
        arr[i] = key;
    }
}
```

Ans 13 =
```
Void bubblesort (int arr[], int n)
{ int i, j;
  bool swapped;
  for (i=0; i<n-1 ; i++)
  {   Swapped = False;
      For ( j=0; j<n-i-1 ; j++)
      { if (arr[j] > arr [j+1])
          { swap (& arr[j], & arr [j+1]);
              Swapped = true;
          }
      }
      if (swapped == false)
        { break;
        }
  }
}
```