ADA-LAB-5

Q) a)Sort a given set of N integer elements using Quick Sort technique and compute its time taken

b)Implement 0/1 Knapsack problem using dynamic programming.

CODE-

Quick sort-

```
#include <stdio.h>
#include <stdlib.h>


void merge(int low,int mid,int high,int array[20],int mer[20])
{
    int i = low;
    int j = mid+1;
    int k = 0;

    while(i<=mid && j<=high)
    {
        if(array[i]<array[j])
        {
            mer[k] = array[i];
            i++;
            k++;
        }
        else
        {
            mer[k] = array[j];
            j++;
            k++;
        }
    }

    while (i <= mid)
    {
        mer[k] = array[i];
        i++;
        k++;
    }

    while (j <= high)
    {
        mer[k] = array[j];
        j++;
        k++;
    }

    for(int i=0;i<k;i++)
    {
        array[low+i] = mer[i];
    }
}
```

```
void merge_sort(int low,int high,int array[20],int merged[20])
{
    if(low<high)
    {
        int mid = (low+high)/2;
        merge_sort(low,mid,array,merged);
        merge_sort(mid+1,high,array,merged);
        merge(low,mid,high,array,merged);
    }
}


int main()
{
    int n,array[30];
    printf("Enter no of elements:");
    scanf("%d",&n);
printf("Enter elements:");

    for(int i=0;i<n;i++)
    {
        scanf("%d",&array[i]);
    }

    int merged[30];

merge_sort(0,n-1,array,merged);
printf("Sorted array:");

    for(int i=0;i<n;i++)
    {
        printf("%d ",array[i]);
    }
}
#include <stdio.h>


void swap(int *a, int *b) {
  int t = *a;
  *a = *b;
  *b = t;
}


int partition(int a[], int l, int h) {
  int pivot = a[l];
  int i = l, j = h;


  while (i < j) {
    while (a[i] <= pivot && i <= h) {
      i++;
    }
    while (a[j] > pivot) {
      j--;
    }
    if (i < j) {
      swap(&a[i], &a[j]);
    }
  }
  swap(&a[l], &a[j]);

  return j;
}

void quickSort(int a[], int l, int h) {
  if (l < h) {
    int pi = partition(a, l, h);
    quickSort(a, l, pi - 1);
    quickSort(a, pi + 1, h);
  }
}

int main() {
  int a[20], n, i;
  printf("Enter size of array\n");
  scanf("%d", &n);
  printf("Enter data elements: ");
  for (i = 0; i < n; i++) {
    scanf("%d", &a[i]);
  }

  printf("Unsorted Array\n");
  for (i = 0; i < n; i++) {
    printf("%d\t", a[i]);
  }

  quickSort(a, 0, n - 1);

  printf("\nSorted array in ascending order: \n");
  for (i = 0; i < n; i++) {
    printf("%d\t", a[i]);
  }
  return 0;
```

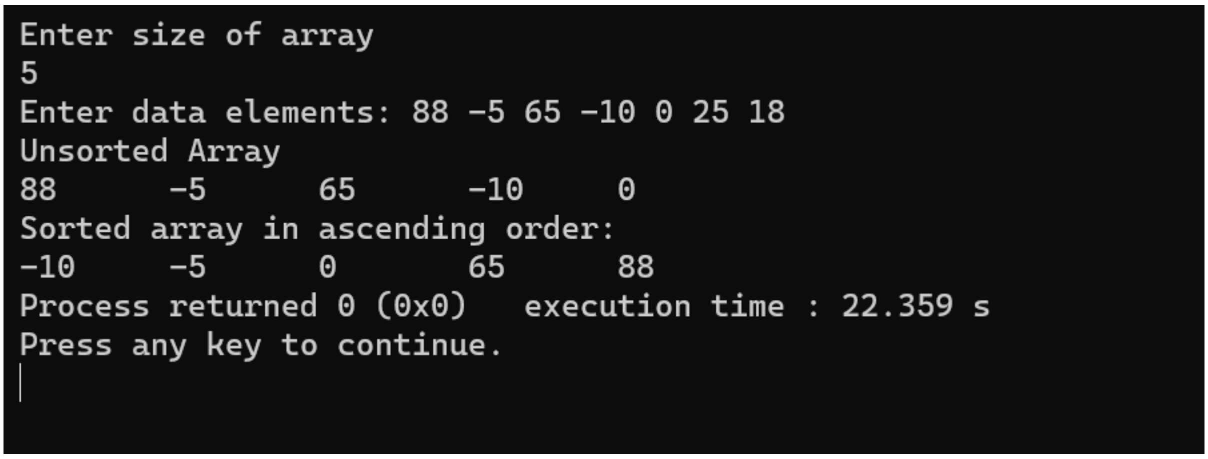
```
}
```

Knapsack problem

```
#include <stdio.h>



int knap(int w[], int p[], int n, int ww) {
int v[n+1][ww+1];

    for (int i = 0; i < n + 1; i++) {
        for (int j = 0; j < ww + 1; j++) {
            if (i == 0 || j == 0) {
                v[i][j] = 0;
                continue;
            } else {
                if (w[i - 1] > j) {
                    v[i][j] = v[i - 1][j];
                } else {
                    if (v[i - 1][j] > (v[i - 1][j - w[i - 1]] + p[i - 1])) {
                        v[i][j] = v[i - 1][j];
                    } else {
                        v[i][j] = v[i - 1][j - w[i - 1]] + p[i - 1];
                    }
                }
            }
        }
    }

    int q = v[n][ww];
    return q;
}

int main() {
    int w[10], p[10], n, ww, ans;
    printf("Enter the number of items: ");
    scanf("%d", &n);
    printf("Enter the weight and profit of each item:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d %d", &w[i], &p[i]);
    }
    printf("Enter the required weight limit: ");
    scanf("%d", &ww);
    ans = knap(w, p, n, ww);
    printf("Maximum profit: %d\n", ans);
    return 0;
}
```

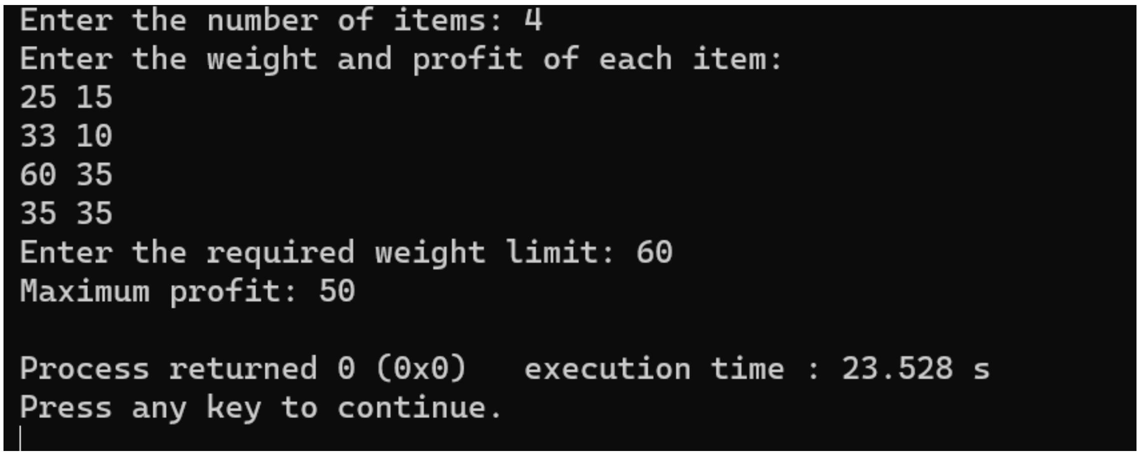
OUTPUT-

QUICK SORT-



```
Enter size of array
5
Enter data elements: 88 -5 65 -10 0 25 18
Unsorted Array
88      -5      65      -10     0
Sorted array in ascending order:
-10     -5      0       65      88
Process returned 0 (0x0)   execution time : 22.359 s
Press any key to continue.
```

KNAPSACK



```
Enter the number of items: 4
Enter the weight and profit of each item:
25 15
33 10
60 35
35 35
Enter the required weight limit: 60
Maximum profit: 50

Process returned 0 (0x0)   execution time : 23.528 s
Press any key to continue.
```