DESIGN AND ANALYSIS OF ALOGORITHM

Question :

Write a C program to sort a given set of elements using the Quicksort method and determine the time required to sort the elements. Write the output. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n.

Answer :

```
1. #include<stdio.h>
2. #include <time.h>
3. #include <stdlib.h>
4. void swap(int* a, int* b)
5. {
6. int t = *a;
7. *a = *b;
8. *b = t;
9. }
10. int partition (int arr[], int low, int high)
11. {
12. int j;
13. int pivot = arr[high];
14. int i = (low - 1);
15. for ( j = low; j <= high- 1; j++)
16. {
17. if (arr[j] <= pivot)
18. {
19. i++;
20. swap(&arr[i], &arr[j]);
21. }
22. }
23. swap(&arr[i + 1], &arr[high]);
24. return (i + 1);
25. }
26. void quickSort(int arr[], int low, int high)
```

```
{
if (low < high)
{
int pi = partition(arr, low, high);
quickSort(arr, low, pi - 1);
quickSort(arr, pi + 1, high);
}
}
void printArray(int arr[], int size)
{
int i;
for (i=0; i < size; i++)
printf("%d ", arr[i]);
printf("\n");
}
int main()
{
time_t start,end;
double tc;
int arr[100000],k,c;
int n = sizeof(arr)/sizeof(arr[0]);
printf("Enter the number of elements :");
scanf("%i",&n);
for(c=n;c>0;c--)
{
arr[c]=rand();
}
start=clock();
quickSort(arr, 0, n-1);
end=clock();
tc=(difftime(end,start)/CLOCKS_PER_SEC);
printf("Total time required : %lf",tc);
return 0;
}
```

OUTPUT :

1. Enter the number of elements : 3000
   Total time required : 0.000313

2. Enter the number of elements : 3500
   Total time required : 0.000368

3. Enter the number of elements : 4000
   Total time required : 0.000437

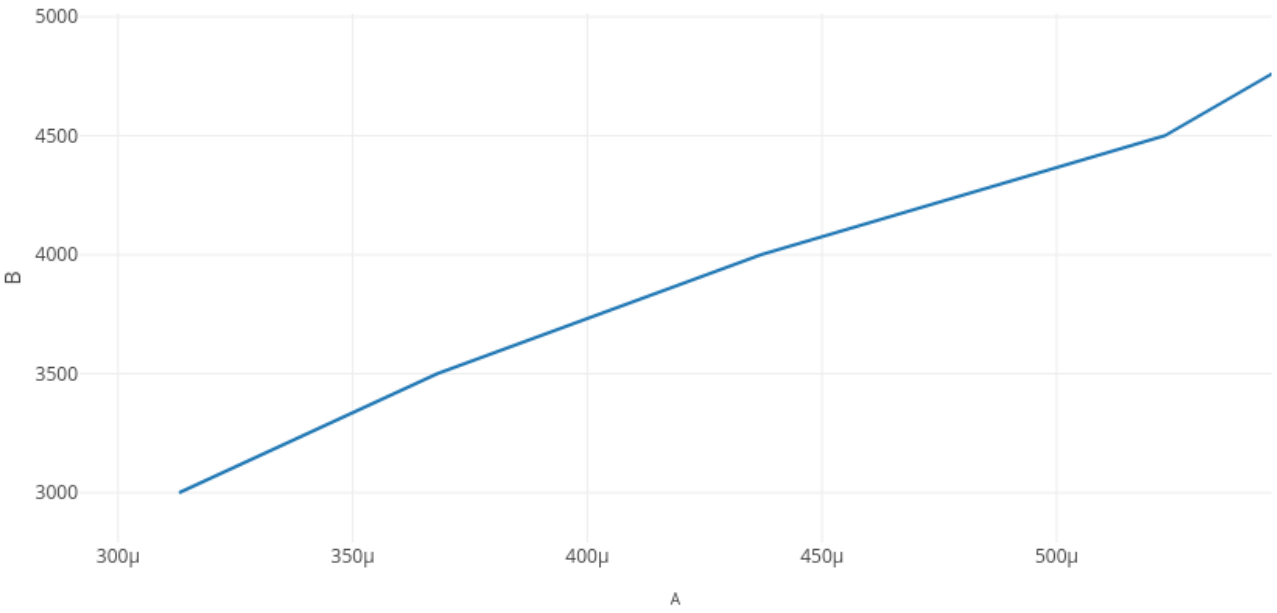
4. Enter the number of elements : 4500
   Total time required : 0.000523

5. Enter the number of elements : 5000
   Total time required : 0.000567

GRAPH :

A = Time required for sorting the list

B = Elements in the list



SUBMITTED BY :

SHILPA O

S6 CSE B

NO: 96