File: conquicksort.cpp

```
#include<iostream>
#include<omp.h>
#include<stdlib.h>
#include<fstream>
using namespace std;
int n;
//stores the no. of elements
int partition(int a[],int p, int r);
//partition the array into two
void quicksort(int a[], int p, int r){
     int q;
     if(p<r){}
          q=partition(a,p,r);
          //partition function returns the pivot element
          #pragma omp parallel sections
          {
               #pragma omp section
                    quicksort(a,p,q-1);
               #pragma omp section
                    quicksort(a,q+1,r);
               }
          }
     }
}
int partition(int a[],int p, int r){
     int i=p-1;
     int x=a[r];
     int temp;
     for(int j=p; j<=r-1; j++){
          if(a[j] \le x){
               i++;
               temp=a[i];
               a[i]=a[j];
               a[j]=temp;
          }
     temp=a[i+1];
     a[i+1]=a[r];
     a[r]=temp;
return i+1;
}
```

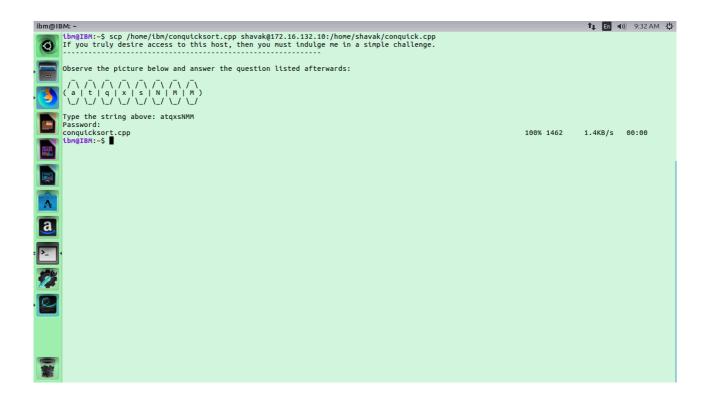
```
int main() {
     ofstream file;
     file.open("time.txt",ios::app);
     cout<<"\n Enter no. of elements : ";</pre>
     cin>>n;
     int a[n];
     for(int i=0; i<n; i++){
        a[i]=rand()%10000;
     cout<<"\n Unsorted array : ";
     for(int i=0; i<n; i++){
          cout<<a[i]<<"\t";
     cout<<"\n";
     double start = omp_get_wtime();
     quicksort(a,0,n-1);
     //invoking the quicksort function
     double finish = omp_get_wtime();
     cout<<"\n Sorted array : ";
     for(int i=0; i<n; i++){
          cout<<a[i]<<"\t";
     }
     double time=finish-start;
     file<<"\nInput dataset size : "<<n<<" "<<"Execution time :
"<<time;
     file.close();
cout<<"\n";
return 0;
}
File: serial.cpp
#include<iostream>
#include<omp.h>
#include<stdlib.h>
#include<fstream>
using namespace std;
int n;
//stores the no. of elements
int partition(int a[],int p, int r);
//partition the array into two
void quicksort(int a[], int p, int r)
     if(p<r)
     {
          int q=partition(a,p,r);
          //partition function returns the pivot element
          quicksort(a,p,q-1);
          quicksort(a,q+1,r);
     }
}
```

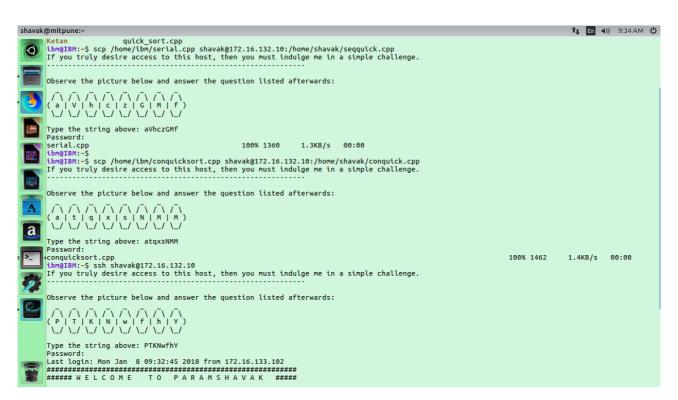
```
int partition(int a[],int p, int r)
{
     int i=p-1;
     int x=a[r];
     int temp;
     for(int j=p; j<=r-1; j++){
          if(a[j] \le x)
               i++;
               temp=a[i];
               a[i]=a[j];
               a[j]=temp;
          }
     temp=a[i+1];
     a[i+1]=a[r];
     a[r]=temp;
return i+1;
}
int main() {
     ofstream file;
     file.open("time.txt",ios::app);
     cout<<"\n Enter no. of elements : ";</pre>
     cin>>n;
     int a[n];
     for(int i=0; i<n; i++){
        a[i]=rand()%10000;
     }
     cout<<"\n Unsorted array : ";
     for(int i=0; i<n; i++){
          cout<<a[i]<<"\t";
     }
     cout<<"\n";
     double start = omp_get_wtime();
     quicksort(a,0,n-1);
     //invoking the quicksort function
     double finish = omp_get_wtime();
     cout<<"\n Sorted array : ";
     for(int i=0; i<n; i++){
          cout<<a[i]<<"\t";
     }
     double time=finish-start;
     file<<"Input dataset size : "<<n<<" "<<"Execution time :
"<<time<<"\n";
     file.close();
cout<<"\n";
return 0;
}
```

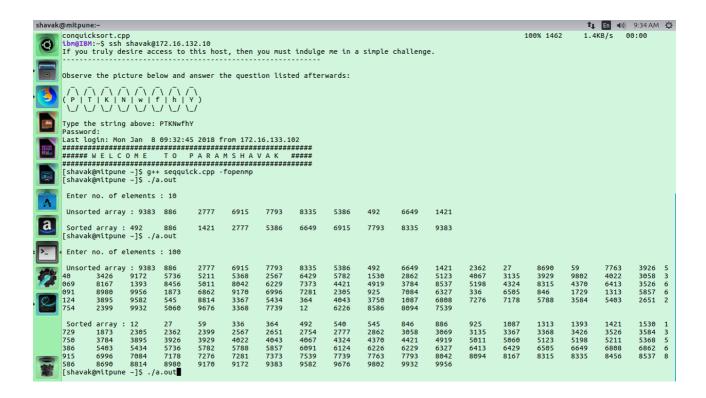
Time Analysis:

```
#Sequencial quicksort:
Input dataset size: 10 Execution time: 1.423e-06
Input dataset size: 100 Execution time: 1.4156e-05
Input dataset size: 1000 Execution time: 0.000184297
Input dataset size: 10000 Execution time: 0.00213747
Input dataset size : 100000 Execution time : 0.0152165
Input dataset size: 1000000 Execution time: 0.369424
#Concurrent quicksort
Input dataset size: 10 Execution time: 0.00322852
Input dataset size: 100 Execution time: 0.00280494
Input dataset size: 1000 Execution time: 0.00358758
Input dataset size: 10000 Execution time: 0.00731108
Input dataset size: 100000 Execution time: 0.0428147
Input dataset size: 1000000 Execution time: 0.681787
#Param Shavak
#Sequencial quicksort
Input dataset size: 10 Execution time: 1.704e-06
Input dataset size: 100 Execution time: 1.7203e-05
Input dataset size: 1000 Execution time: 0.000237504
Input dataset size: 10000 Execution time: 0.00159052
Input dataset size: 100000 Execution time: 0.0221149
Input dataset size: 1000000 Execution time: 0.311625
#Concurrent quicksort
Input dataset size: 10 Execution time: 0.00753268
Input dataset size: 100 Execution time: 0.00792356
Input dataset size: 1000 Execution time: 0.00386392
Input dataset size : 10000 Execution time : 0.0123936
Input dataset size: 100000 Execution time: 0.0561629
Input dataset size: 1000000 Execution time: 0.752313
```

#OUTPUT:

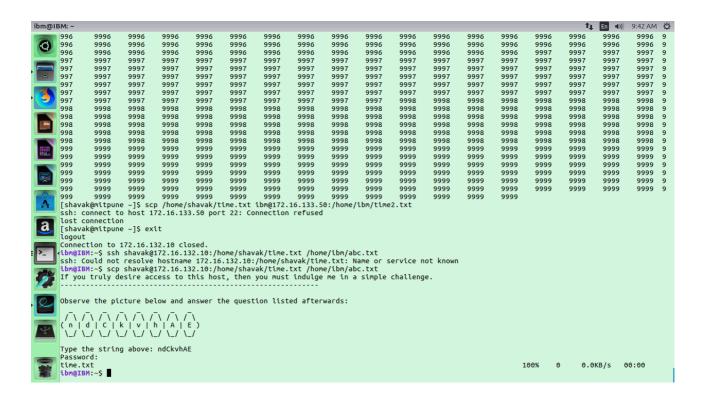






shavak	@mitpun	e:~														1 _‡	En 40)	9:35 AM 以
	091	8980	9956	1873	6862	9170	6996	7281	2305	925	7084	6327	336	6505	846	1729	1313	5857 6
	124	3895	9582	545	8814	3367	5434	364	4043	3750	1087	6808	7276	7178	5788	3584	5403	2651 2
	754	2399	9932	5060	9676	3368	7739	12	6226	8586	8094	7539						
	Contac		12	27	F0	226	264	492	F 4 0	545	0.46	006	025	1007	1212	1202	1421	1530 1
	Sorted 729	l array : 1873	12 2305	2362	59 2399	336 2567	364 2651	2754	540 2777	2862	846 3058	886 3069	925 3135	1087 3367	1313 3368	1393 3426	1421 3526	1530 1 3584 3
	750	3784	3895	3926	3929	4022	4043	4067	4324	4370	4421	4919	5011	5060	5123	5198	5211	5368 5
	386	5403	5434	5736	5782	5788	5857	6091	6124	6226	6229	6327	6413	6429	6505	6649	6808	6862 6
	915	6996	7084	7178	7276	7281	7373	7539	7739	7763	7793	8042	8094	8167	8315	8335	8456	8537 8
	586	8690	8814	8980	9170	9172	9383	9582	9676	9802	9932	9956						
	1	@mitpune																
		ed array		886	2777	6915	7793	8335	5386	492	6649	1421	2362	27	8690	59	7763	3926 5
	40	3426	9172	5736	5211	5368	2567	6429	5782	1530	2862	5123	4067	3135	3929	9802	4022	3058 3
	069	8167	1393	8456	5011	8042	6229	7373	4421	4919	3784	8537	5198	4324	8315	4370	6413	3526 6
	091 124	8980 3895	9956 9582	1873 545	6862 8814	9170 3367	6996 5434	7281 364	2305 4043	925 3750	7084 1087	6327 6808	336 7276	6505 7178	846 5788	1729 3584	1313 5403	5857 6 2651 2
لثا	754	2399	9932	5060	9676	3368	7739	12	6226	8586	8094	7539	795	570	1434	378	7467	6601 9
-0-	7	2902	3317	492	6652	756	7301	280	4286	9441	3865	9689	8444	6619	8440	4729	8031	8117 8
A	097	5771	4481	675	709	8927	4567	7856	9497	2353	4586	6965	5306	4683	6219	8624	1528	2871 5
	732	8829	9503	19	8270	3368	9708	6715	6340	8149	7796	723	2618	2245	2846	3451	2921	3555 2
	379	7488	7764	8228	9841	2350	5193	1500	7034	7764	124	4914	6987	5856	3743	6491	2227	8365 9
a	859	1936	1432	2551	6437	9228	3275	5407	1474	6121	8858	4395	6029	1237	8235	3793	5818	4428 6
	143	1011	5928	9529	8776	2404	4443	5763	4613	4538	8606	6840	2904	4818	5128	688	7369	7917 9
	917	6996	3324	7743	9470	2183	8490	5499	9772	6725	5644	5590	7505	8139	2954	9786	7669	8082 8
: >_	4542 605	8464	197 5661	9507	9355	8804	6348	8611 8736	3622 9444	7828	9299	7343	5746	5568	4340	5422	3311 2924	3810 7 7637 2
	062	1801 5624	2600	3730 2036	4878 3452	1305 1899	9320 9379	5550	7468	8626 71	8522 973	3465 7131	6708 3881	3416 4930	8282 8933	3258 5894	8660	7637 2 163 7
100	199	7981	8899	2996	2959	3773	2813	9668	7190	1095	2926	6466	5084	1340	2090	7684	3376	5542 5
	936	9107	7445	9756	9179	8418	6887	9412	3348	2172	1659	2009	2336	5210	6342	7587	8206	9301 7
	713	7372	5321	1255	4819	4599	7721	9904	5939	9811	3940	5667	1705	6228	1127	9150	5984	6658 3
	920	9224	2422	7269	1396	4081	5630	84	9292	1972	7672	3850	7625	5385	1222	9299	6640	6042 3
	898	713	2298	6190	524	2590	8209	8581	8819	9336	7732	1155	5994	8004	379	4769	5273	1776 8
	850	7255	1860	8142	5579	5884	1993	3205	7621	9567	2504	613	1961	2754	1326	4259	8944	8202 3
	202	3506	6784	2021	2842	868	9528	5189	8872	9908	9958	498	8036	8808	7753	6248	3303	3333 2
	133	1648	2890	9754	7567	1746	368	9529	4500	8046	3788	9797	6249	6990	3303	3033	5363	2497 2
	53	4892	7686	9125	1152	3996	5975	9188	9157	3729	5436	2460	3414	3921	460	6304	28	8027 8
	050 535	6748	7556	8902	4794	7697	8699	1043	1039	2002	428	6403	4500	681	7647	8538	6159	5151 2 7971 6
	949	2134 289	4339 5367	1692 7988	2215 2292	6127 5795	504 743	5629 3144	49 2829	964 8390	8285 1682	6429 5340	5343 3541	6335 569	3177 3826	2900 4232	5238 2261	7971 6 6042 3
188	60	9117	8023	6761	81	6309	743 3190	5425	8996	6367	4677	4234	690	1626	4524	6057	9614	3168 8
2003	00	7111	0023	0701	01	0309	3190	5723	0990	0307	7077	7234	090	1020	7324	0037	5014	2100 0

shaval	@mitpune	:~														1 _L	En 40)	9:36 AM	₩.
0			~]\$ g++ ~]\$./a		ck.cpp -	fopenmp													
			lements																
ت	Unsort	ed array	: 9383	886	2777	6915	7793	8335	5386	492	6649	1421							
3		array : @mitpune	492 ~]\$./a	886 .out	1421	2777	5386	6649	6915	7793	8335	9383							
	Enter	no. of e	lements	: 100															
		ed array		886	2777	6915	7793	8335	5386	492	6649	1421	2362	27	8690	59	7763	3926	
	40 069	3426 8167	9172 1393	5736 8456	5211 5011	5368 8042	2567 6229	6429 7373	5782 4421	1530 4919	2862 3784	5123 8537	4067 5198	3135 4324	3929 8315	9802 4370	4022 6413	3058 3526	3
	091	8980	9956	1873	6862	9170	6996	7281	2305	925	7084	6327	336	6505	846	1729	1313		6
	124	3895	9582	545	8814	3367	5434	364	4043	3750	1087	6808	7276	7178	5788	3584	5403	2651	2
	754	2399	9932	5060	9676	3368	7739	12	6226	8586	8094	7539							
-	Sorted		12	27 2362	59 2399	336 2567	364	492 2754	540 2777	545	846 3058	886	925	1087	1313	1393	1421 3526	1530 3584	
A	729 750	1873 3784	2305 3895	3926	3929	4022	2651 4043	4067	4324	2862 4370	4421	3069 4919	3135 5011	3367 5060	3368 5123	3426 5198	5211	5368	
	386	5403	5434	5736	5782	5788	5857	6091	6124	6226	6229	6327	6413	6429	6505	6649	6808	6862	
	915	6996	7084	7178	7276	7281	7373	7539	7739	7763	7793	8042	8094	8167	8315	8335	8456	8537	8
a,	586 [shavak	8690 @mitpune	8814 ~]\$./a	8980 .out	9170	9172	9383	9582	9676	9802	9932	9956							
: >_	• Enter	no. of e	lements	: 1000															
4261		ed array		886	2777	6915	7793	8335	5386	492	6649	1421	2362	27	8690	59	7763	3926	
1	40 069	3426	9172	5736	5211	5368	2567	6429	5782	1530	2862	5123	4067	3135	3929	9802	4022	3058	
	009	8167 8980	1393 9956	8456 1873	5011 6862	8042 9170	6229 6996	7373 7281	4421 2305	4919 925	3784 7084	8537 6327	5198 336	4324 6505	8315 846	4370 1729	6413 1313	3526 5857	
	124	3895	9582	545	8814	3367	5434	364	4043	3750	1087	6808	7276	7178	5788	3584	5403	2651	
	754	2399	9932	5060	9676	3368	7739	12	6226	8586	8094	7539	795	570	1434	378	7467	6601	
	7	2902	3317	492	6652	756	7301	280	4286	9441	3865	9689	8444	6619	8440	4729	8031	8117	
	097 732	5771 8829	4481 9503	675 19	709 8270	8927 3368	4567 9708	7856 6715	9497 6340	2353 8149	4586 7796	6965 723	5306 2618	4683 2245	6219 2846	8624 3451	1528 2921	2871 3555	2
	379	7488	7764	8228	9841	2350	5193	1500	7034	7764	124	723 4914	6987	5856	3743	6491	2921	8365	
	859	1936	1432	2551	6437	9228	3275	5407	1474	6121	8858	4395	6029	1237	8235	3793	5818		6
	143	1011	5928	9529	8776	2404	4443	5763	4613	4538	8606	6840	2904	4818	5128	688	7369	7917	
18th	917	6996	3324	7743	9470	2183	8490	5499	9772	6725	5644	5590	7505	8139	2954	9786	7669		8
30	542	8464	197	9507	9355	8804	6348	8611	3622	7828	9299	7343	5746	5568	4340	5422	3311	3810	7



ibm@IB	BM: ~															1 _‡	En ◀)))	10:08 AM 😃
		l:~\$ g++ l:~\$./a.		ksort.cp	p -fopen	mp												
	Enter	no. of e	lements	: 10														
	Unsort	ed array	: 9383	886	2777	6915	7793	8335	5386	492	6649	1421						
		array : :~\$./a.		886	1421	2777	5386	6649	6915	7793	8335	9383						
	Enter	no. of e	lements	: 100														
	Unsort 40 069 091 124 754	ed array 3426 8167 8980 3895 2399	: 9383 9172 1393 9956 9582 9932	886 5736 8456 1873 545 5060	2777 5211 5011 6862 8814 9676	6915 5368 8042 9170 3367 3368	7793 2567 6229 6996 5434 7739	8335 6429 7373 7281 364 12	5386 5782 4421 2305 4043 6226	492 1530 4919 925 3750 8586	6649 2862 3784 7084 1087 8094	1421 5123 8537 6327 6808 7539	2362 4067 5198 336 7276	27 3135 4324 6505 7178	8690 3929 8315 846 5788	59 9802 4370 1729 3584	7763 4022 6413 1313 5403	3926 5 3058 3 3526 6 5857 6 2651 2
a	729 750 386 915 586	1873 3784 5403 6996 8690	2305 3895 5434 7084 8814	27 2362 3926 5736 7178 8980	59 2399 3929 5782 7276 9170	336 2567 4022 5788 7281 9172	364 2651 4043 5857 7373 9383	492 2754 4067 6091 7539 9582	540 2777 4324 6124 7739 9676	545 2862 4370 6226 7763 9802	846 3058 4421 6229 7793 9932	886 3069 4919 6327 8042 9956	925 3135 5011 6413 8094	1087 3367 5060 6429 8167	1313 3368 5123 6505 8315	1393 3426 5198 6649 8335	1421 3526 5211 6808 8456	1530 1 3584 3 5368 5 6862 6 8537 8
>		no. of e		: 1000														
	Unsort 40 069 091 124 754 7 097 732 379 859 143 917 542	3426 8167 8980 3895 2399 2902 5771 8829 7488 1936 1011 6996 8464	: 9383 9172 1393 9956 9582 9932 3317 4481 9503 7764 1432 5928 3324	886 5736 8456 1873 545 5060 492 675 19 8228 2551 9529 7743 9507	2777 5211 5011 6862 8814 9676 6652 709 8270 9841 6437 8776 9470 9355	6915 5368 8042 9170 3367 3368 756 8927 3368 2350 9228 2404 2183 8804	7793 2567 6229 6996 5434 7739 7301 4567 9708 5193 3275 4443 8490 6348	8335 6429 7373 7281 364 12 280 7856 6715 1500 5407 5763 5499 8611	5386 5782 4421 2305 4043 6226 4286 9497 6340 7034 1474 4613 9772 3622	492 1530 4919 925 3750 8586 9441 2353 8149 7764 6121 4538 6725 7828	6649 2862 3784 7084 1087 8094 3865 4586 7796 124 8858 8606 5644 9299	1421 5123 8537 6327 6808 7539 9689 6965 723 4914 4395 6840 5590 7343	2362 4067 5198 336 7276 795 8444 5306 2618 6987 6029 2904 7505 5746	27 3135 4324 6505 7178 570 6619 4683 2245 5856 1237 4818 8139 5568	8690 3929 8315 846 5788 1434 8440 6219 2846 3743 8235 5128 2954 4340	59 9802 4370 1729 3584 378 4729 8624 3451 6491 3793 688 9786 5422	7763 4022 6413 1313 5403 7467 8031 1528 2921 2227 5818 7369 7669 3311	3926 5 3058 3 3526 6 5857 6 2651 2 6601 9 8117 8 2871 5 3555 2 8365 2 8462 8 7917 9 8082 8

CPU Usage:

