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]<=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 : ";
     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

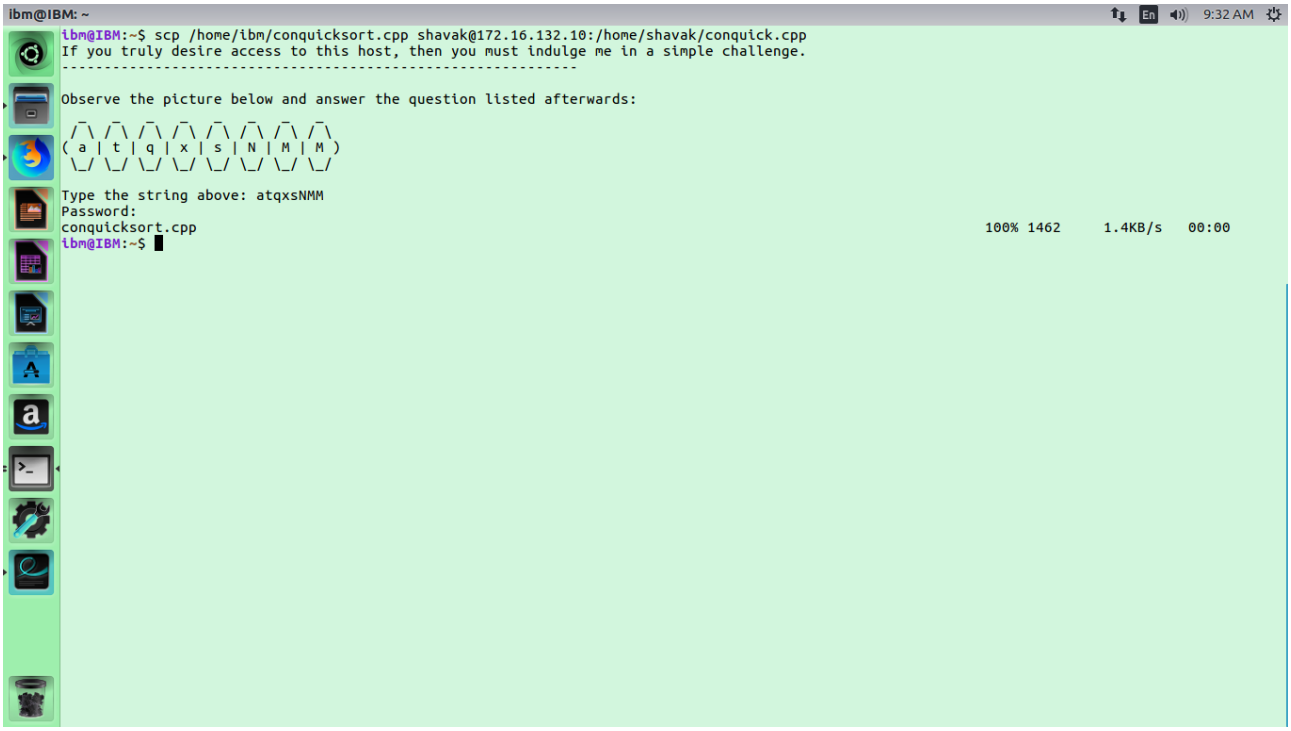
```
#include<iostream>
#include<omp.h>
#include<stdlib.h>
#include<fstream>
using namespace std;
int n;
//stores the no. of elements
int partition(int a[],int p, int r);
//partition the array into two

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);
     }
}
```

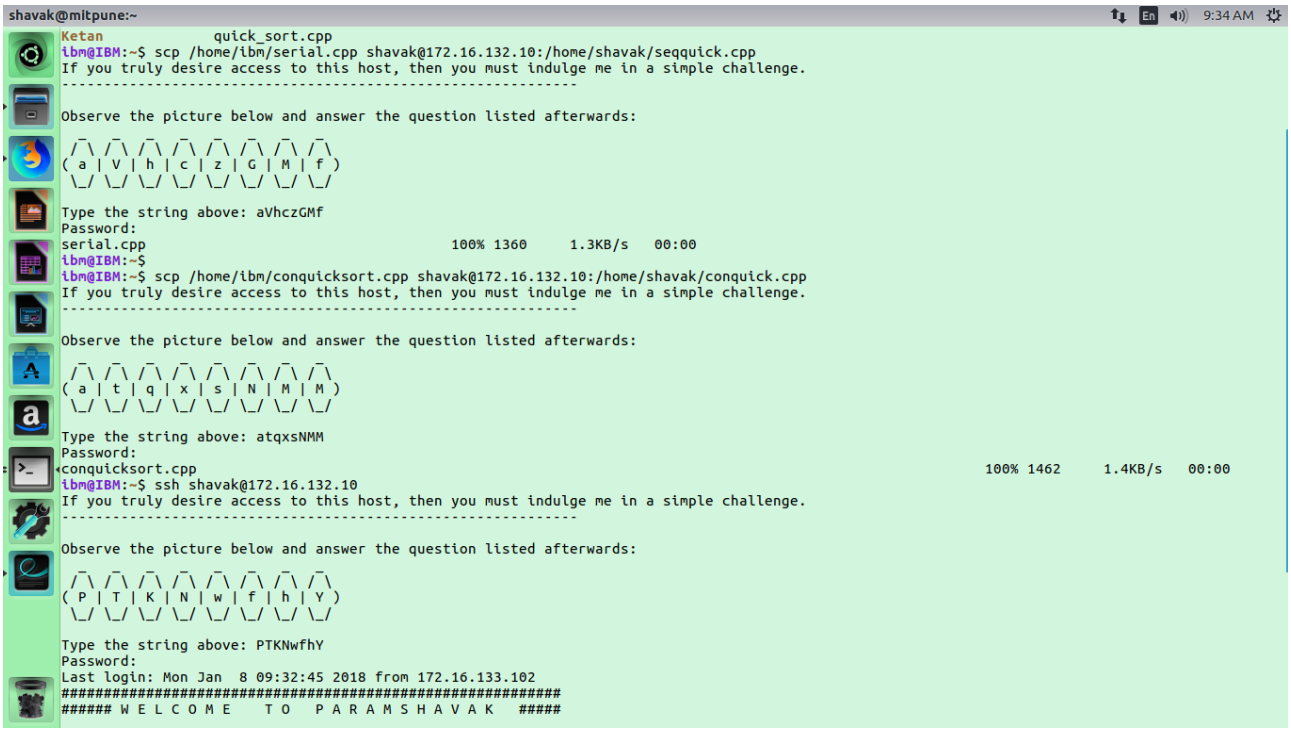
```
int partition(int a[],int p, int r)
{
     int i=p-1;
     int x=a[r];
     int temp;

     for(int j=p; j<=r-1; j++){
          if(a[j]<=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 : ";
     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;
}
```

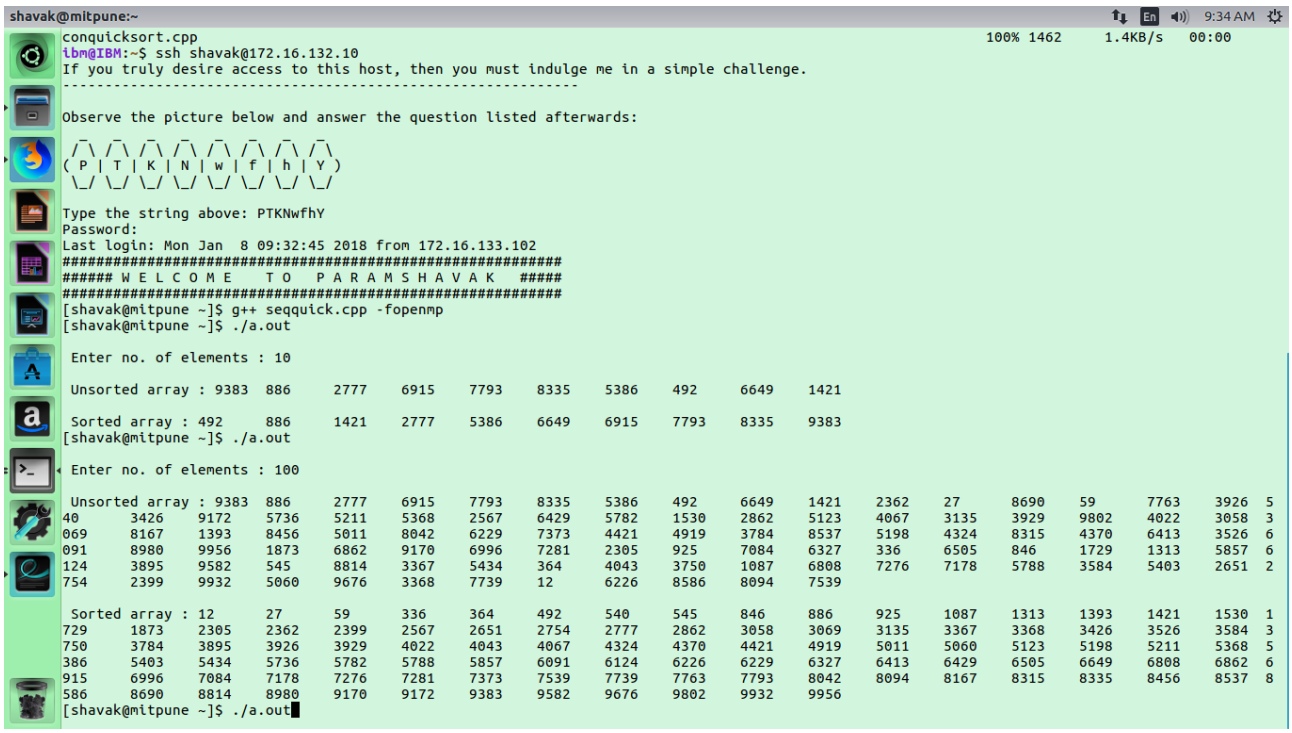
Time Analysis:

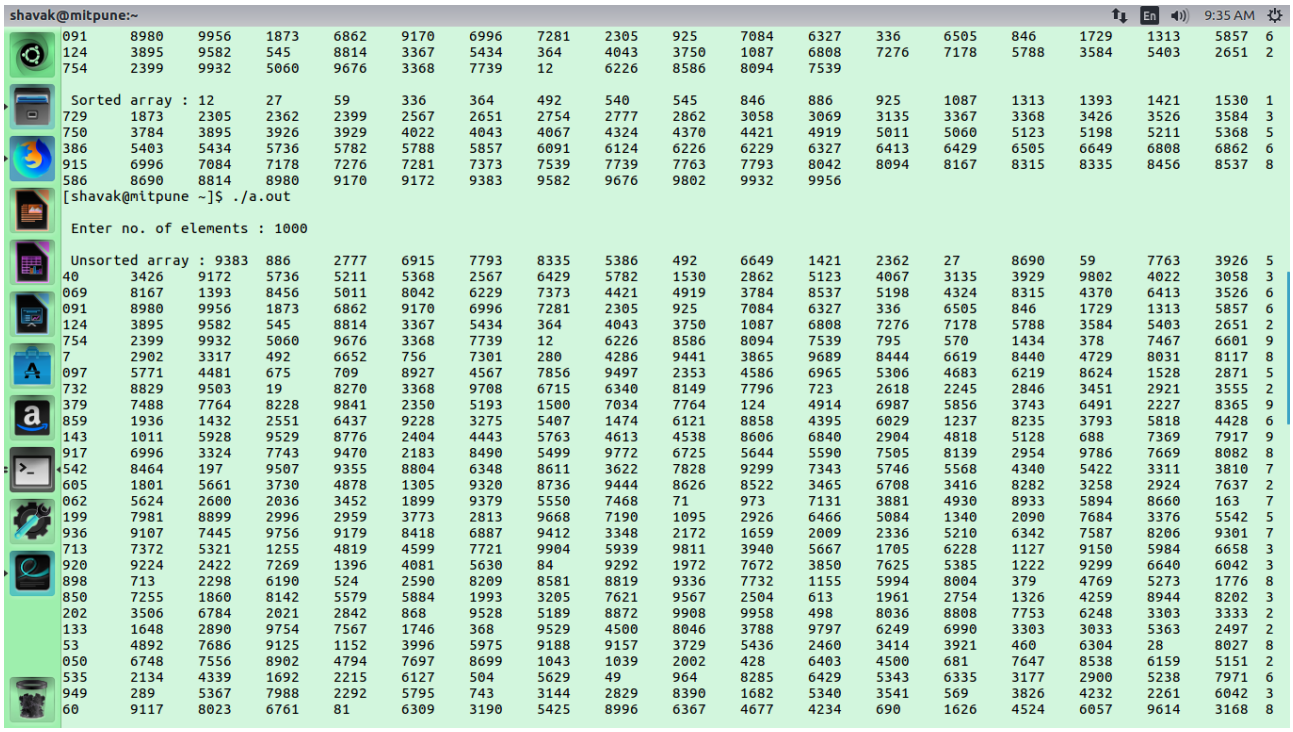
```
#Sequencial quicksort:
Input dataset size : 10  Execution time : 1.423e-06
Input dataset size : 100  Execution time : 1.4156e-05
Input dataset size : 1000  Execution time : 0.000184297
Input dataset size : 10000  Execution time : 0.00213747
Input dataset size : 100000  Execution time : 0.0152165
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
```

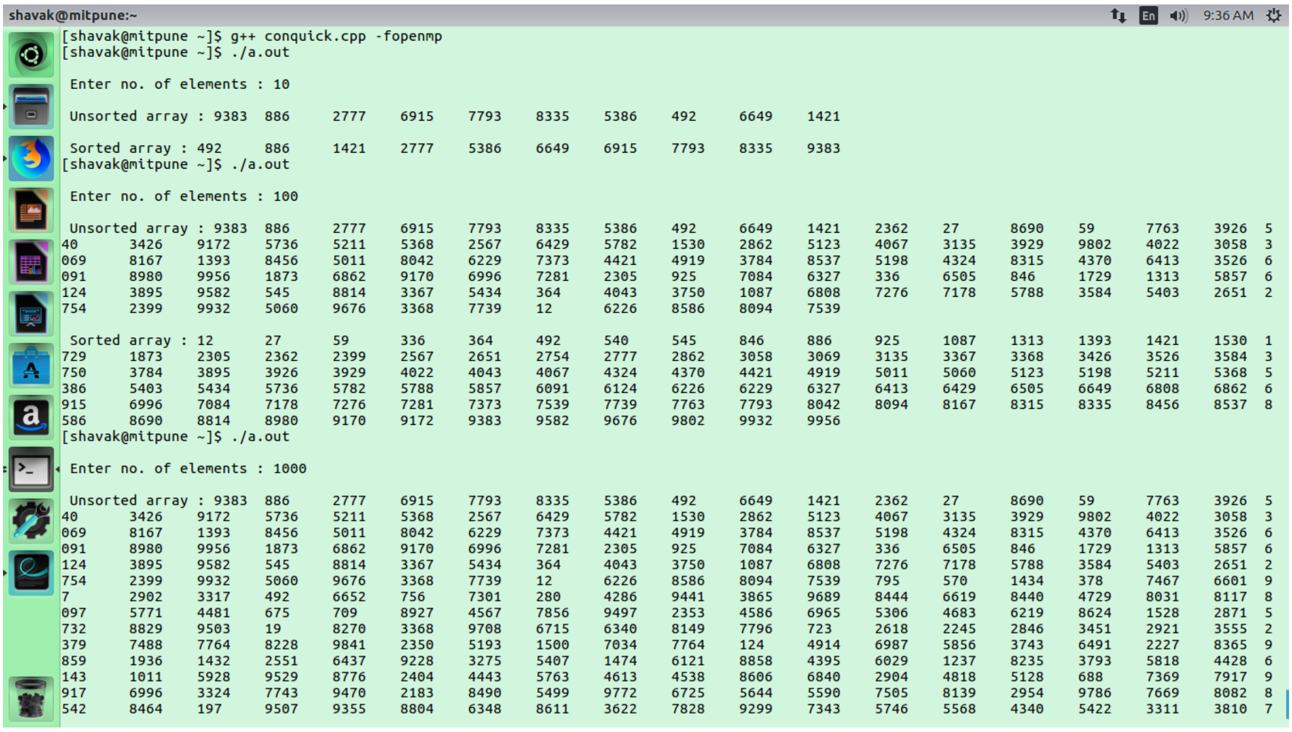
#OUTPUT:


ibm@IBM: ~
9:32 AM
ibm@IBM:~$ scp /home/ibm/conquicksort.cpp shavak@172.16.132.10:/home/shavak/conquick.cpp
If you truly desire access to this host, then you must indulge me in a simple challenge.
Observe the picture below and answer the question listed afterwards:
(a | t | q | x | s | N | M | M)
Type the string above: atqxsNMM
Password:
conquicksort.cpp 100% 1462 1.4KB/s 00:00
ibm@IBM:~$

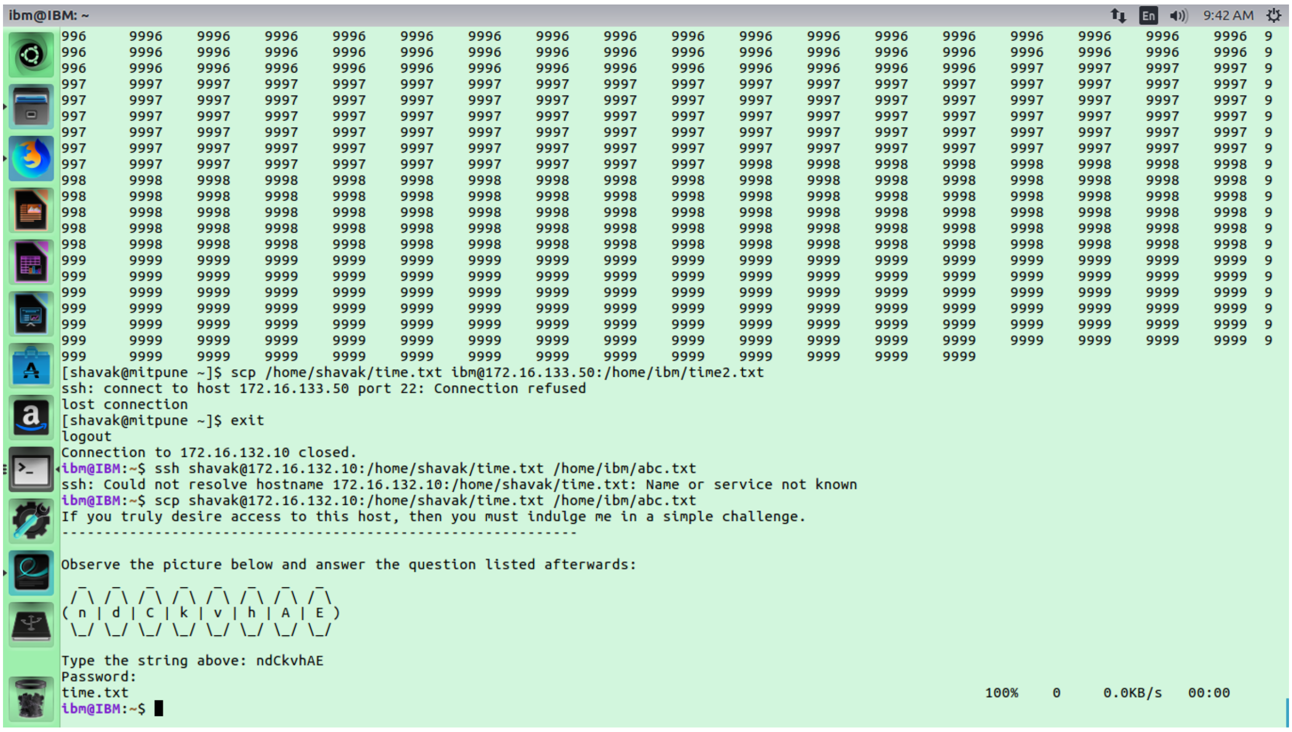

shavak@mitpune:~
9:34 AM
Ketan quick_sort.cpp
ibm@IBM:~$ scp /home/ibm/serial.cpp shavak@172.16.132.10:/home/shavak/seqquick.cpp
If you truly desire access to this host, then you must indulge me in a simple challenge.
Observe the picture below and answer the question listed afterwards:
(a | V | h | c | z | G | M | f)
Type the string above: aVhczGMf
Password:
serial.cpp 100% 1360 1.3KB/s 00:00
ibm@IBM:~$
ibm@IBM:~$ scp /home/ibm/conquicksort.cpp shavak@172.16.132.10:/home/shavak/conquick.cpp
If you truly desire access to this host, then you must indulge me in a simple challenge.
Observe the picture below and answer the question listed afterwards:
(a | t | q | x | s | N | M | M)
Type the string above: atqxsNMM
Password:
conquicksort.cpp 100% 1462 1.4KB/s 00:00
ibm@IBM:~$ ssh shavak@172.16.132.10
If you truly desire access to this host, then you must indulge me in a simple challenge.
Observe the picture below and answer the question listed afterwards:
(P | T | K | N | w | f | h | Y)
Type the string above: PTKNwfhY
Password:
Last login: Mon Jan 8 09:32:45 2018 from 172.16.133.102
W E L C O M E T O P A R A M S H A V A K


shavak@mitpune:~
9:34 AM
conquicksort.cpp 100% 1462 1.4KB/s 00:00
ibm@IBM:~$ ssh shavak@172.16.132.10
If you truly desire access to this host, then you must indulge me in a simple challenge.
Observe the picture below and answer the question listed afterwards:
(P | T | K | N | w | f | h | Y)
Type the string above: PTKNwfhY
Password:
Last login: Mon Jan 8 09:32:45 2018 from 172.16.133.102
W E L C O M E T O P A R A M S H A V A K
[shavak@mitpune ~]$ g++ seqquick.cpp -fopenmp
[shavak@mitpune ~]$ ./a.out
Enter no. of elements : 10
Unsorted array : 9383 886 2777 6915 7793 8335 5386 492 6649 1421
Sorted array : 492 886 1421 2777 5386 6649 6915 7793 8335 9383
[shavak@mitpune ~]$ ./a.out
Enter no. of elements : 100
Unsorted array : 9383 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 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
Sorted array : 12 27 59 336 364 492 540 545 846 886 925 1087 1313 1393 1421 1530 1
729 1873 2305 2362 2399 2567 2651 2754 2777 2862 3058 3069 3135 3367 3368 3426 3526 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
[shavak@mitpune ~]$ ./a.out


shavak@mitpune:~
9:35 AM
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
Sorted array : 12 27 59 336 364 492 540 545 846 886 925 1087 1313 1393 1421 1530 1
729 1873 2305 2362 2399 2567 2651 2754 2777 2862 3058 3069 3135 3367 3368 3426 3526 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
[shavak@mitpune ~]$ ./a.out
Enter no. of elements : 1000
Unsorted array : 9383 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 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 795 570 1434 378 7467 6601 9
7 2902 3317 492 6652 756 7301 280 4286 9441 3865 9689 8444 6619 8440 4729 8031 8117 8
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
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
542 8464 197 9507 9355 8804 6348 8611 3622 7828 9299 7343 5746 5568 4340 5422 3311 3810 7
605 1801 5661 3730 4878 1305 9320 8736 9444 8626 8522 3465 6708 3416 8282 3258 2924 7637 2
062 5624 2600 2036 3452 1899 9379 5550 7468 71 973 7131 3881 4930 8933 5894 8660 163 7
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 6748 7556 8902 4794 7697 8699 1043 1039 2002 428 6403 4500 681 7647 8538 6159 5151 2
535 2134 4339 1692 2215 6127 504 5629 49 964 8285 6429 5343 6335 3177 2900 5238 7971 6
949 289 5367 7988 2292 5795 743 3144 2829 8390 1682 5340 3541 569 3826 4232 2261 6042 3
60 9117 8023 6761 81 6309 3190 5425 8996 6367 4677 4234 690 1626 4524 6057 9614 3168 8



```
shavak@mitpune:~                                                                                      9:36 AM
[shavak@mitpune ~]$ g++ conquick.cpp -fopenmp
[shavak@mitpune ~]$ ./a.out

 Enter no. of elements : 10

 Unsorted array : 9383  886     2777    6915    7793    8335    5386    492     6649    1421

 Sorted array : 492     886     1421    2777    5386    6649    6915    7793    8335    9383
[shavak@mitpune ~]$ ./a.out

 Enter no. of elements : 100

 Unsorted array : 9383  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     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

 Sorted array : 12      27      59      336     364     492     540     545     846     886     925     1087    1313    1393    1421    1530    1
729     1873    2305    2362    2399    2567    2651    2754    2777    2862    3058    3069    3135    3367    3368    3426    3526    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
[shavak@mitpune ~]$ ./a.out

 Enter no. of elements : 1000

 Unsorted array : 9383  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     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    795     570     1434    378     7467    6601    9
7       2902    3317    492     6652    756     7301    280     4286    9441    3865    9689    8444    6619    8440    4729    8031    8117    8
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
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
542     8464    197     9507    9355    8804    6348    8611    3622    7828    9299    7343    5746    5568    4340    5422    3311    3810    7
```



```
ibm@IBM: ~                                                                                            9:42 AM
996     9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9
996     9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9
996     9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9996    9997    9997    9997    9997    9
997     9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9
997     9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9
997     9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9
997     9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9
997     9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9997    9
997     9997    9997    9997    9997    9997    9997    9997    9997    9997    9998    9998    9998    9998    9998    9998    9998    9998    9
998     9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9
998     9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9
998     9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9
998     9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9
998     9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9998    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9
999     9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999    9999
[shavak@mitpune ~]$ scp /home/shavak/time.txt ibm@172.16.133.50:/home/ibm/time2.txt
ssh: connect to host 172.16.133.50 port 22: Connection refused
lost connection
[shavak@mitpune ~]$ exit
logout
Connection to 172.16.132.10 closed.
ibm@IBM:~$ ssh shavak@172.16.132.10:/home/shavak/time.txt /home/ibm/abc.txt
ssh: Could not resolve hostname 172.16.132.10:/home/shavak/time.txt: Name or service not known
ibm@IBM:~$ scp shavak@172.16.132.10:/home/shavak/time.txt /home/ibm/abc.txt
If you truly desire access to this host, then you must indulge me in a simple challenge.
--------------------------------------------------------------

Observe the picture below and answer the question listed afterwards:
  _   _   _   _   _   _   _   _
 / \ / \ / \ / \ / \ / \ / \ / \
( n | d | C | k | v | h | A | E )
 \_/ \_/ \_/ \_/ \_/ \_/ \_/ \_/

Type the string above: ndCkvhAE
Password:
time.txt                                                                  100%    0     0.0KB/s   00:00
ibm@IBM:~$
```

```
ibm@IBM:~$ g++ conquicksort.cpp -fopenmp
ibm@IBM:~$ ./a.out

 Enter no. of elements : 10

 Unsorted array : 9383  886     2777    6915    7793    8335    5386    492     6649    1421

 Sorted array : 492     886     1421    2777    5386    6649    6915    7793    8335    9383
ibm@IBM:~$ ./a.out

 Enter no. of elements : 100

 Unsorted array : 9383  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     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

 Sorted array : 12      27      59      336     364     492     540     545     846     886     925     1087    1313    1393    1421    1530  1
729     1873    2305    2362    2399    2567    2651    2754    2777    2862    3058    3069    3135    3367    3368    3426    3526    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
ibm@IBM:~$ ./a.out

 Enter no. of elements : 1000

 Unsorted array : 9383  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     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    795     570     1434    378     7467    6601  9
7       2902    3317    492     6652    756     7301    280     4286    9441    3865    9689    8444    6619    8440    4729    8031    8117  8
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
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
542     8464    197     9507    9355    8804    6348    8611    3622    7828    9299    7343    5746    5568    4340    5422    3311    3810  7
```

CPU Usage: