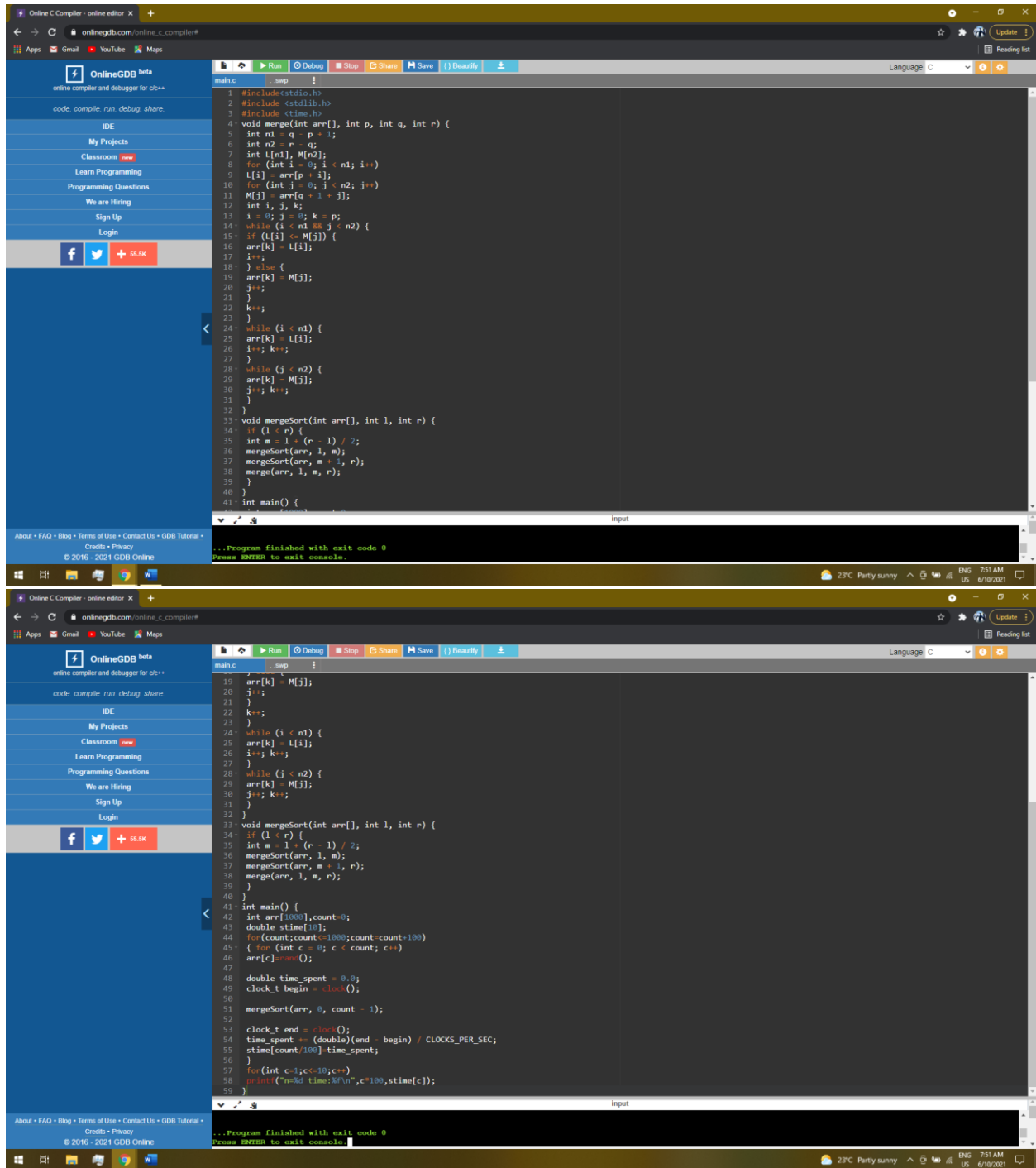


OUTPUT FOR QUICKSORT



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 void merge(int arr[], int p, int q, int r) {
5     int n1 = q - p + 1;
6     int n2 = r - q;
7     int l[n1], M[n2];
8     for (int i = 0; i < n1; i++)
9         l[i] = arr[p + i];
10    for (int j = 0; j < n2; j++)
11        M[j] = arr[q + 1 + j];
12    int i, j, k;
13    i = 0; j = 0; k = p;
14    while (i < n1 && j < n2) {
15        if (l[i] <= M[j]) {
16            arr[k] = l[i];
17            i++;
18        } else {
19            arr[k] = M[j];
20            j++;
21        }
22        k++;
23    }
24    while (i < n1) {
25        arr[k] = l[i];
26        i++; k++;
27    }
28    while (j < n2) {
29        arr[k] = M[j];
30        j++; k++;
31    }
32 }
33 void mergeSort(int arr[], int l, int r) {
34     if (l < r) {
35         int m = l + (r - l) / 2;
36         mergeSort(arr, l, m);
37         mergeSort(arr, m + 1, r);
38         merge(arr, l, m, r);
39     }
40 }
41 int main() {
42     // ... (code continues) ...
43 }
```

...Program finished with exit code 0
Press ENTER to exit console.

The screenshot shows the OnlineGDB website interface. On the left is a navigation menu with links like 'My Projects', 'Classrooms', and 'Learn Programming'. The main area displays a C program with two nested loops. Below the code, the 'Input' tab shows the execution results for array sizes from 100 to 1000, including timing data. The program finished with exit code 0.

```

18 int main() {
19     int n;
20     int arr[k] = M[j];
21     int i;
22     int k;
23     while (i < n1) {
24         arr[k] = L[i];
25         i++; k++;
26     }
27     while (j < n2) {
28         arr[k] = M[j];
29         j++; k++;
30     }
31 }

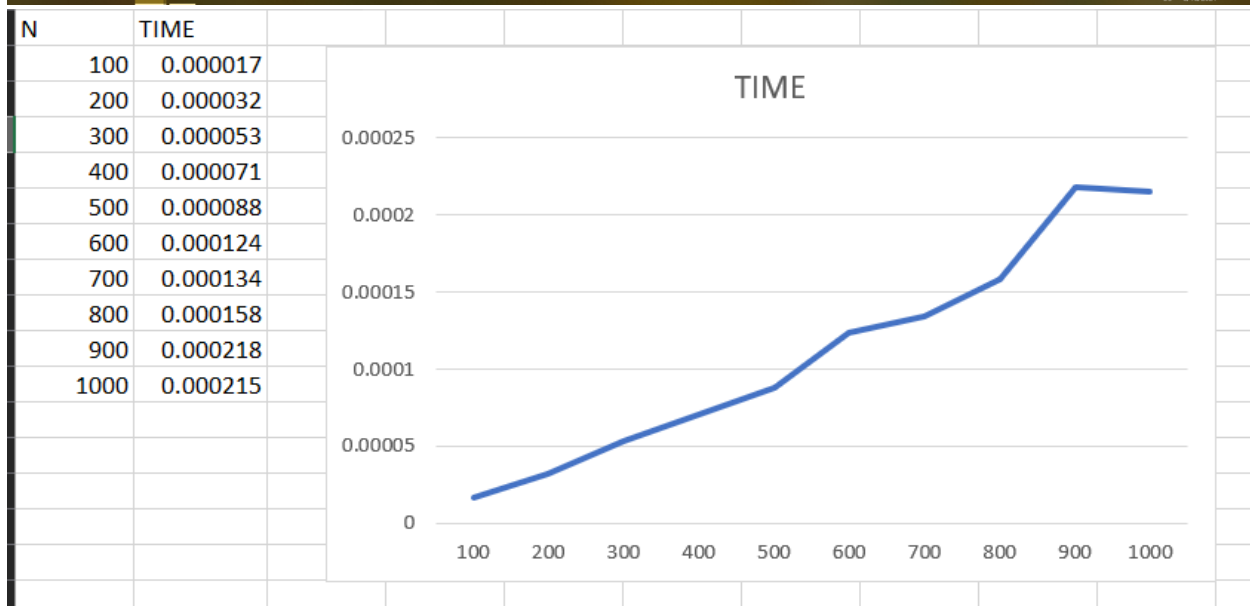
```

Input:

```

n=100 time:0.000018
n=200 time:0.000035
n=300 time:0.000055
n=400 time:0.000071
n=500 time:0.000117
n=600 time:0.000122
n=700 time:0.000144
n=800 time:0.000163
n=900 time:0.000184
n=1000 time:0.000199
...Program finished with exit code 0
Press ENTER to exit console.

```



CODE FOR QUICKSORT

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

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

```
#include <time.h>
```

```
void merge(int arr[], int p, int q, int r) {
```

```
    int n1 = q - p + 1;
```

```

int n2 = r - q;
int L[n1], M[n2];
for (int i = 0; i < n1; i++)
    L[i] = arr[p + i];
for (int j = 0; j < n2; j++)
    M[j] = arr[q + 1 + j];
int i, j, k;
i = 0; j = 0; k = p;
while (i < n1 && j < n2) {
    if (L[i] <= M[j]) {
        arr[k] = L[i];
        i++;
    } else {
        arr[k] = M[j];
        j++;
    }
    k++;
}
while (i < n1) {
    arr[k] = L[i];
    i++; k++;
}
while (j < n2) {
    arr[k] = M[j];
    j++; k++;
}
}

void mergeSort(int arr[], int l, int r) {
    if (l < r) {

```

```

int m = l + (r - l) / 2;
mergeSort(arr, l, m);
mergeSort(arr, m + 1, r);
merge(arr, l, m, r);
}
}

int main() {
    int arr[1000],count=0;
    double stime[10];
    for(count;count<=1000;count=count+100)
    { for (int c = 0; c < count; c++)
        arr[c]=rand();

        double time_spent = 0.0;
        clock_t begin = clock();

        mergeSort(arr, 0, count - 1);

        clock_t end = clock();
        time_spent += (double)(end - begin) / CLOCKS_PER_SEC;
        stime[count/100]=time_spent;
    }
    for(int c=1;c<=10;c++)
        printf("n=%d time:%f\n",c*100,stime[c]);
}

```

OUTPUT FOR MERGESORT

Online C Compiler - online editor

onlinegdb.com/online_c_compiler#

Apps Gmail YouTube Maps

OnlineGDB beta
online compiler and debugger for C/C++
code compile run debug share
IDE
My Projects
Classrooms
Learn Programming
Programming Questions
We are Hiring
Sign Up
Login

Facebook

Twitter

Plus

About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial •
Credits • Privacy
© 2016 - 2021 GDB Online

main.c

Run

Debug

Stop

Share

Save

Fullscreen

Language: C

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 void quicksort(int number[25],int first,int last){
5     int i, j, pivot, temp;
6     if(first<last){
7         pivot=first;
8         i=first;
9         j=last;
10        while(i<j){
11            while(number[i]<=number[pivot]||i<last)
12                i++;
13            while(number[j]>=number[pivot])
14                j--;
15            if(i<j){
16                temp=number[i];
17                number[i]=number[j];
18                number[j]=temp;
19            }
20        }
21        temp=number[pivot];
22        number[pivot]=number[j];
23        number[j]=temp;
24        quicksort(number,first,j-1);
25        quicksort(number,j+1,last);
26    }
27 }
28 int main(){
29     int i, count=0, number[1000];
30     double time[10];
31
32     for(count;count<1000;count=count+100)
33     { for (int c = 0; c < count; c++)
34         number[c]=rand();
35     }
```

Input

Output

```
n=300 time:0.000024
n=400 time:0.000042
n=500 time:0.000042
n=600 time:0.000050
n=700 time:0.000039
n=800 time:0.000049
n=900 time:0.000079
n=1000 time:0.000088

...Program finished with exit code 0
Press ENTER to exit console.
```

Windows Taskbar

23°C Partly sunny 7:54 AM 6/10/2021

Online C Compiler - online editor

onlinegdb.com/online_c_compiler#

Apps Gmail YouTube Maps

OnlineGDB beta
online compiler and debugger for C/C++
code compile run debug share
IDE
My Projects
Classrooms
Learn Programming
Programming Questions
We are Hiring
Sign Up
Login

Facebook

Twitter

Plus

About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial •
Credits • Privacy
© 2016 - 2021 GDB Online

main.c

Run

Debug

Stop

Share

Save

Fullscreen

Language: C

```
19 }
20 }
21 temp=number[pivot];
22 number[pivot]=number[j];
23 number[j]=temp;
24 quicksort(number,first,j-1);
25 quicksort(number,j+1,last);
26 }
27 }
28 int main(){
29     int i, count=0, number[1000];
30     double time[10];
31
32     for(count;count<1000;count=count+100)
33     { for (int c = 0; c < count; c++)
34         number[c]=rand();
35     }
36     double time_spent = 0.0;
37     clock_t begin = clock();
38
39     quicksort(number,0,count-1);
40
41     clock_t end = clock();
42     time_spent += (double)(end - begin) / CLOCKS_PER_SEC;
43     time[count/100]=time_spent;
44 }
45
46
47 for(int c=1;c<10;c++){
48     printf("n=%d time:%f\n",c*100,time[c]);
49 }
50 return 0;
51 }
52 }
```

Input

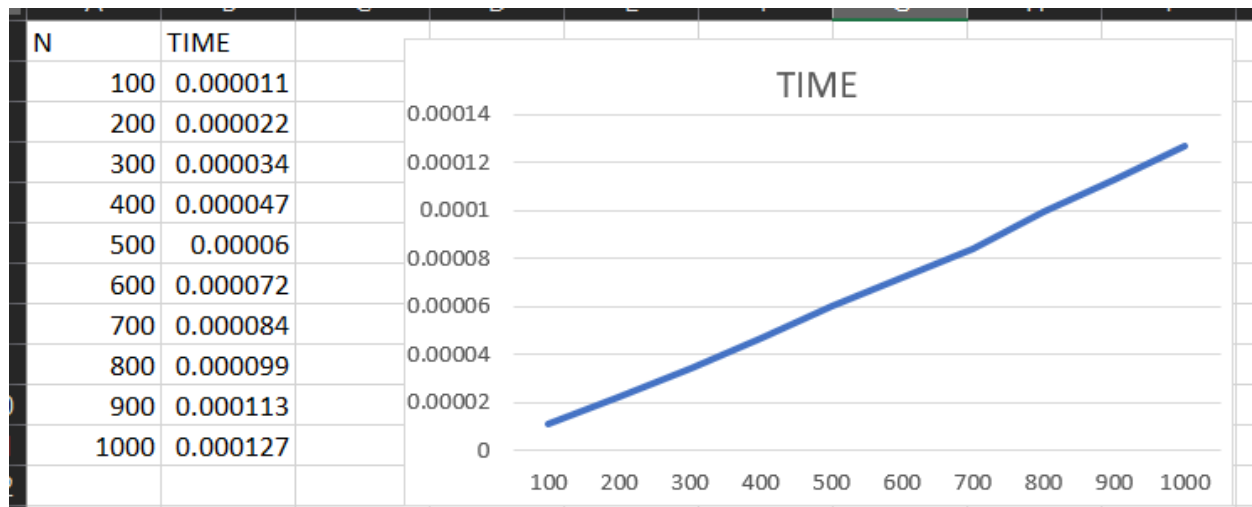
Output

```
n=300 time:0.000024
n=400 time:0.000042
n=500 time:0.000042
n=600 time:0.000050
n=700 time:0.000039
n=800 time:0.000049
n=900 time:0.000079
n=1000 time:0.000088

...Program finished with exit code 0
Press ENTER to exit console.
```

Windows Taskbar

23°C Partly sunny 7:54 AM 6/10/2021



CODE FOR MERGESORT

```
#include<stdio.h>

#include <stdlib.h>

#include <time.h>

void quicksort(int number[25],int first,int last){

    int i, j, pivot, temp;

    if(first<last){

        pivot=first;

        i=first;

        j=last;

        while(i<j){

            while(number[i]<=number[pivot]&& i<last)

                i++;

            while(number[j]>number[pivot])

                j--;

            if(i<j){

                temp=number[i];

                number[i]=number[j];

                number[j]=temp;

            }

        }

    }

}
```

```

    }
    temp=number[pivot];
    number[pivot]=number[j];
    number[j]=temp;
    quicksort(number,first,j-1);
    quicksort(number,j+1,last);
}
}

int main(){
    int i, count=0, number[1000];
    double stime[10];

    for(count;count<=1000;count=count+100)
    { for (int c = 0; c < count; c++)
        number[c]=rand();

        double time_spent = 0.0;
        clock_t begin = clock();

        quicksort(number,0,count-1);

        clock_t end = clock();
        time_spent += (double)(end - begin) / CLOCKS_PER_SEC;
        stime[count/100]=time_spent;

    }

    for(int c=1;c<=10;c++)
        printf("n=%d time:%f\n",c*100,stime[c]);

```

```
return 0;
```

```
}
```