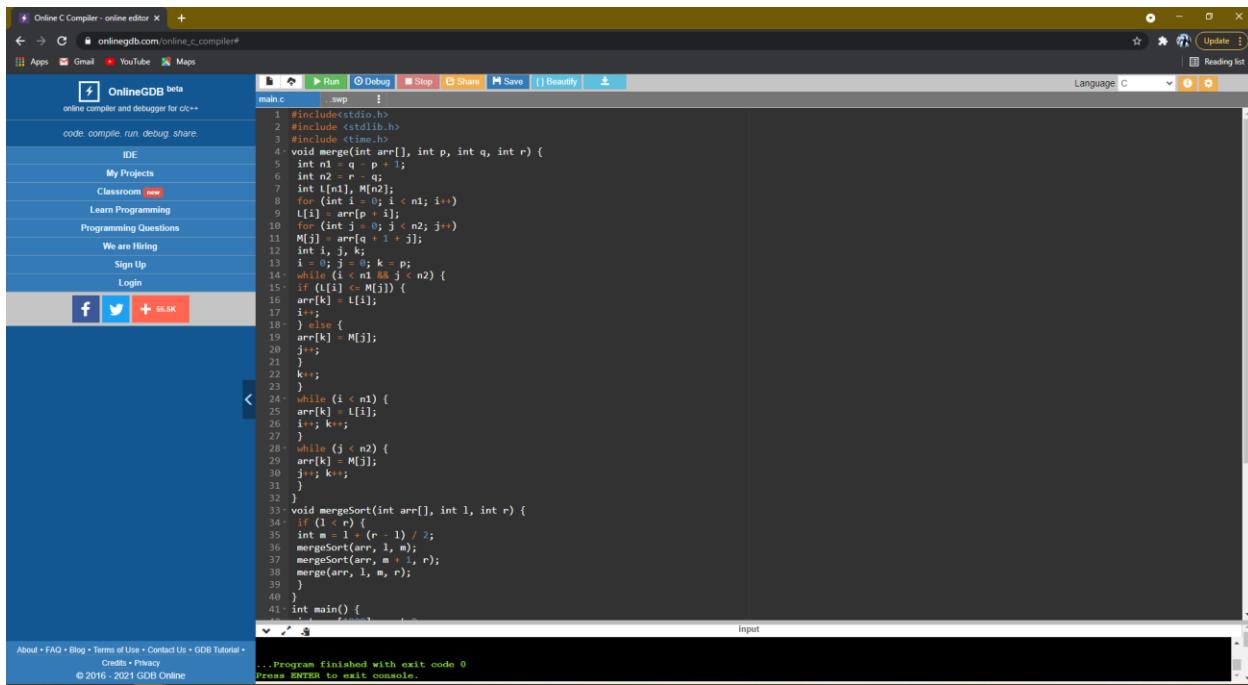


OUTPUT FOR QUICKSORT



OnlineC Compiler - online editor | onlinengdb.com/online_c_compiler#

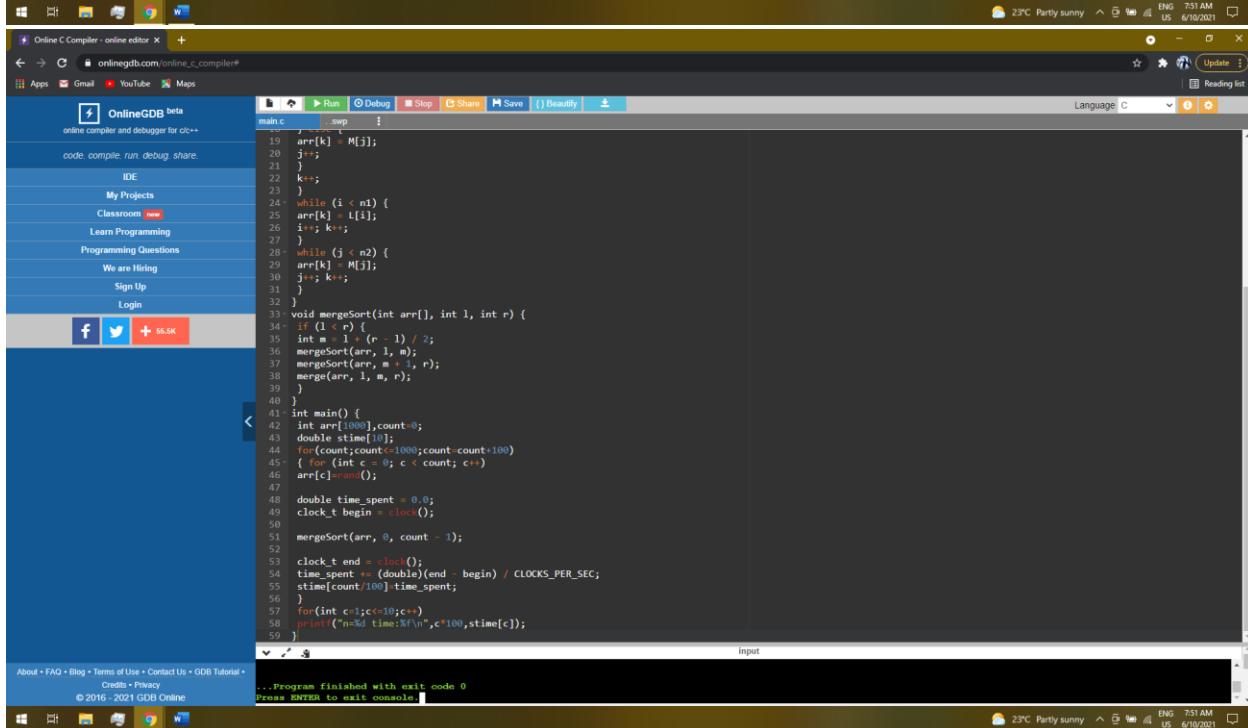
code compile run debug share.

IDE My Projects Classroom Learn Programming Programming Questions We are hiring Sign Up Login

Facebook Twitter + 55.5K

```
main.c    .swp  i
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4
5 void merge(int arr[], int p, int q, int r) {
6     int n1 = q - p + 1;
7     int n2 = r - q;
8     int l[n1], M[n2];
9     for (int i = 0; i < n1; i++)
10    l[i] = arr[p + i];
11    for (int j = 0; j < n2; j++)
12    M[j] = arr[q + 1 + j];
13    int i, j, k;
14    i = 0; j = 0; k = p;
15    while (i < n1 && j < n2) {
16        if (l[i] <= M[j]) {
17            arr[k] = l[i];
18        } else {
19            arr[k] = M[j];
20        }
21        j++;
22        k++;
23    }
24    while (i < n1) {
25        arr[k] = l[i];
26        i++;
27        k++;
28    }
29    while (j < n2) {
30        arr[k] = M[j];
31        j++;
32        k++;
33    }
34}
35 void mergeSort(int arr[], int l, int r) {
36    if (l < r) {
37        int m = l + (r - l) / 2;
38        mergeSort(arr, l, m);
39        mergeSort(arr, m + 1, r);
40    }
41}
42 int main() {
43    int arr[1000], count=0;
44    double stime[10];
45    for(count;count<1000;count=count+100)
46    { for (int c = 0; c < count; c++)
47        arr[c]=rand();
48    }
49    double time_spent = 0.0;
50    clock_t begin = clock();
51    mergeSort(arr, 0, count - 1);
52    clock_t end = clock();
53    time_spent = (double)(end - begin) / CLOCKS_PER_SEC;
54    stime[count/100]=time_spent;
55    for(int c=1;c<10;c++)
56        printf("%d%lf time:%f\n",c*100,stime[c]);
57 }
```

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



OnlineC Compiler - online editor | onlinengdb.com/online_c_compiler#

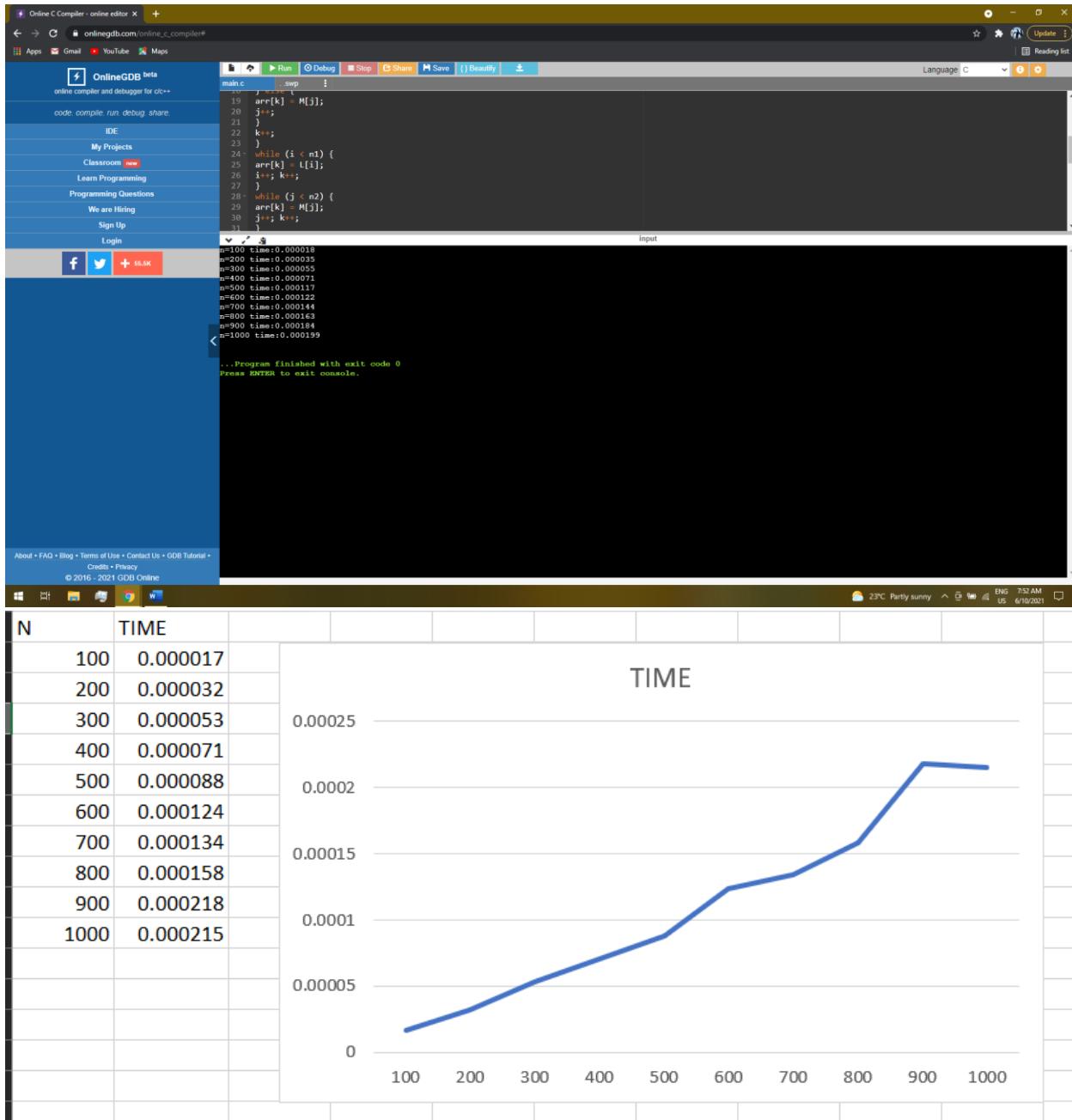
code compile run debug share.

IDE My Projects Classroom Learn Programming Programming Questions We are hiring Sign Up Login

Facebook Twitter + 55.5K

```
main.c    .swp  i
19    arr[k] = M[j];
20    j++;
21    k++;
22}
23}
24}
25 while (i < n1) {
26    arr[k] = l[i];
27    i++;
28    k++;
29}
30 while (j < n2) {
31    arr[k] = M[j];
32    j++;
33}
34}
35 void mergeSort(int arr[], int l, int r) {
36    if (l < r) {
37        int m = l + (r - l) / 2;
38        mergeSort(arr, l, m);
39        mergeSort(arr, m + 1, r);
40    }
41}
42 int main() {
43    int arr[1000],count=0;
44    double stime[10];
45    for(count;count<1000;count=count+100)
46    { for (int c = 0; c < count; c++)
47        arr[c]=rand();
48    }
49    double time_spent = 0.0;
50    clock_t begin = clock();
51    mergeSort(arr, 0, count - 1);
52    clock_t end = clock();
53    time_spent = (double)(end - begin) / CLOCKS_PER_SEC;
54    stime[count/100]=time_spent;
55    for(int c=1;c<10;c++)
56        printf("%d%lf time:%f\n",c*100,stime[c]);
57 }
```

...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 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 X

onlinegdb.com/online_c_compiler#

code compile run debug share.

IDE

My Projects

Classroom

Learn Programming

Programming Questions

We are Hiring

Sign Up

Login

Facebook Twitter LinkedIn

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

n>300 time:0.000024
n>400 time:0.000062
n>500 time:0.000042
n>600 time:0.000050
n>700 time:0.000059
n>800 time:0.000069
n>900 time:0.000079
n>1000 time:0.000088

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

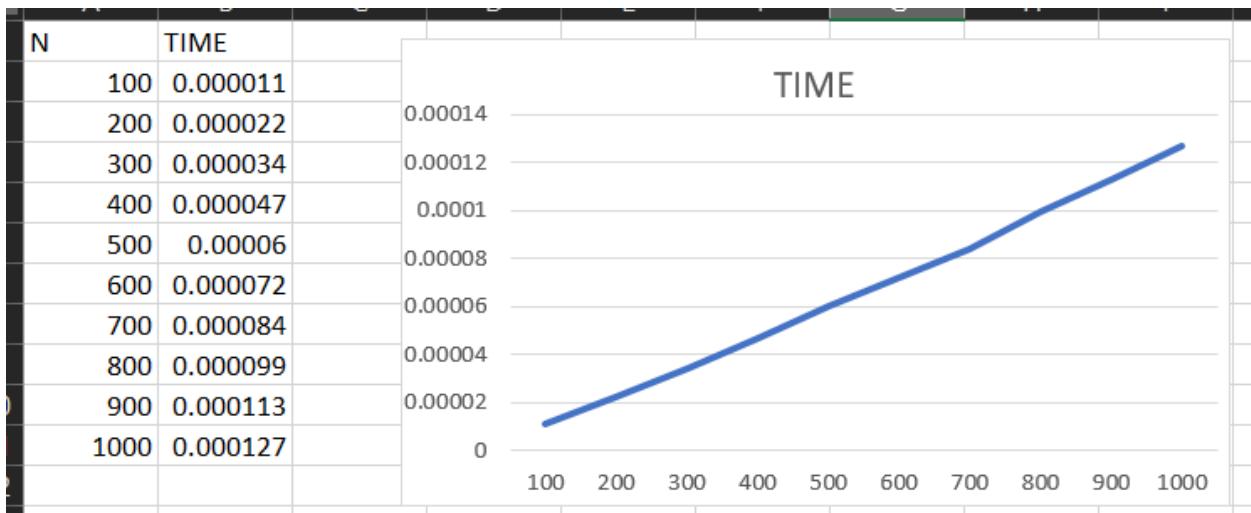
Language: C

Reading list

23°C Party sunny

ENG 7:54 AM

US 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;
```

```
}
```