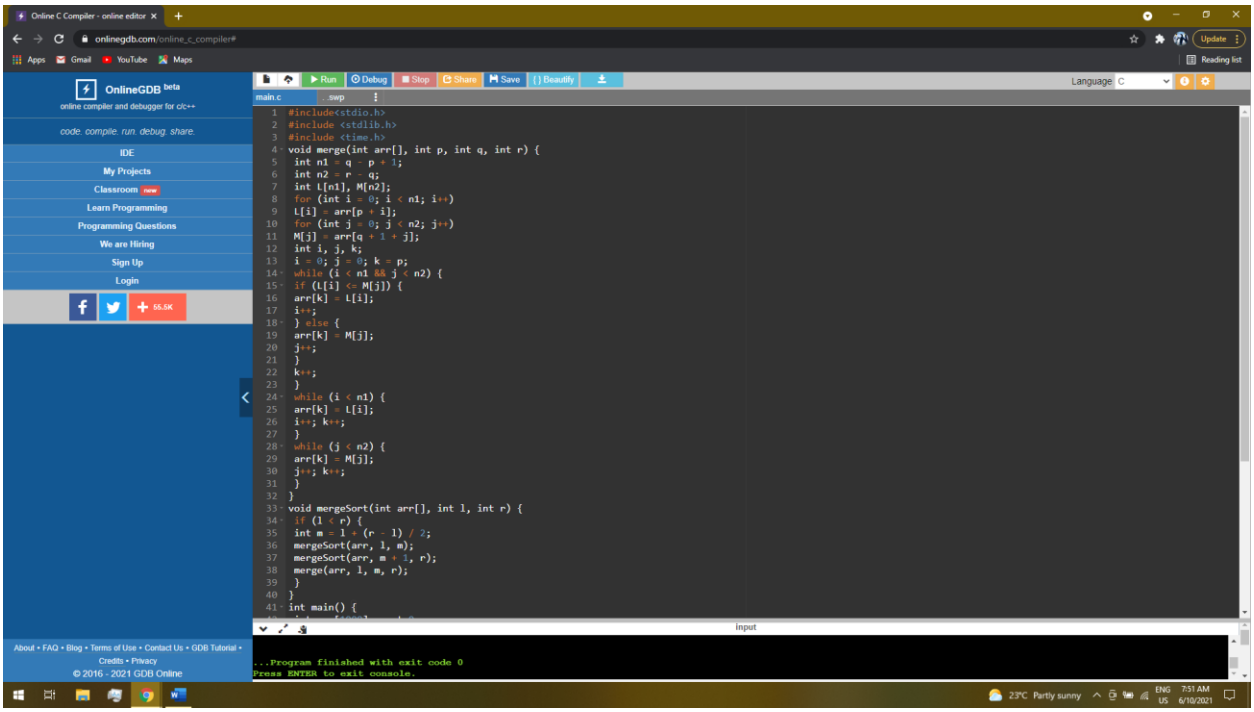
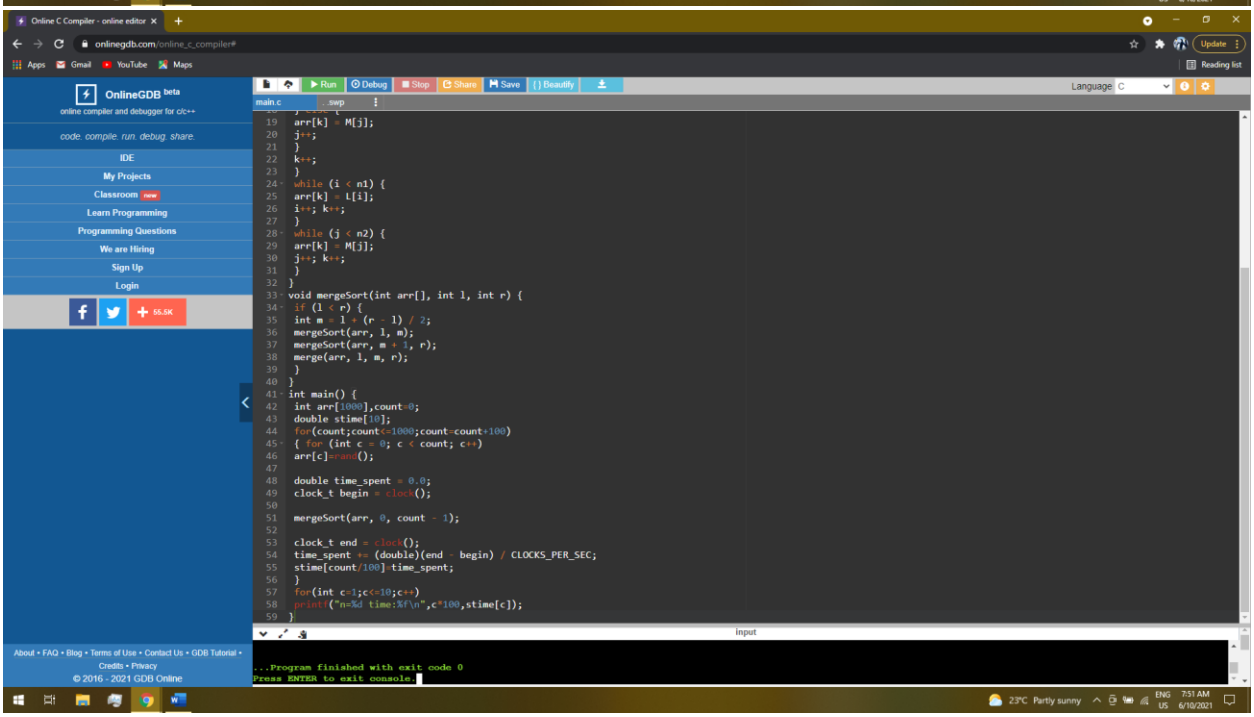
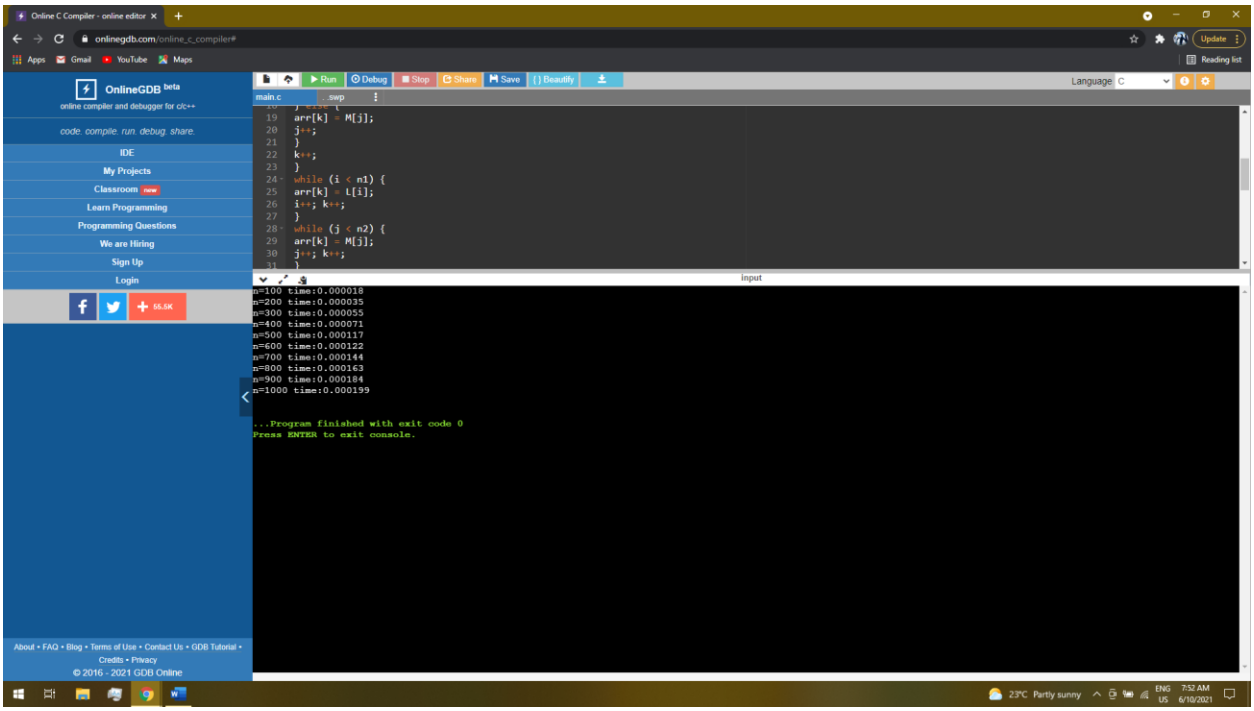
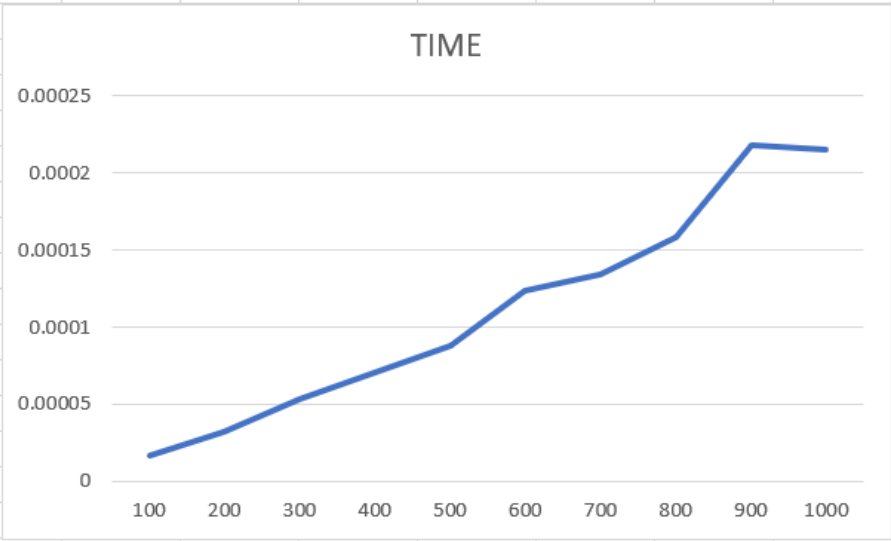
OUTPUT FOR QUICKSORT







N	TIME
100	0.000017
200	0.000032
300	0.000053
400	0.000071
500	0.000088
600	0.000124
700	0.000134
800	0.000158
900	0.000218
1000	0.000215



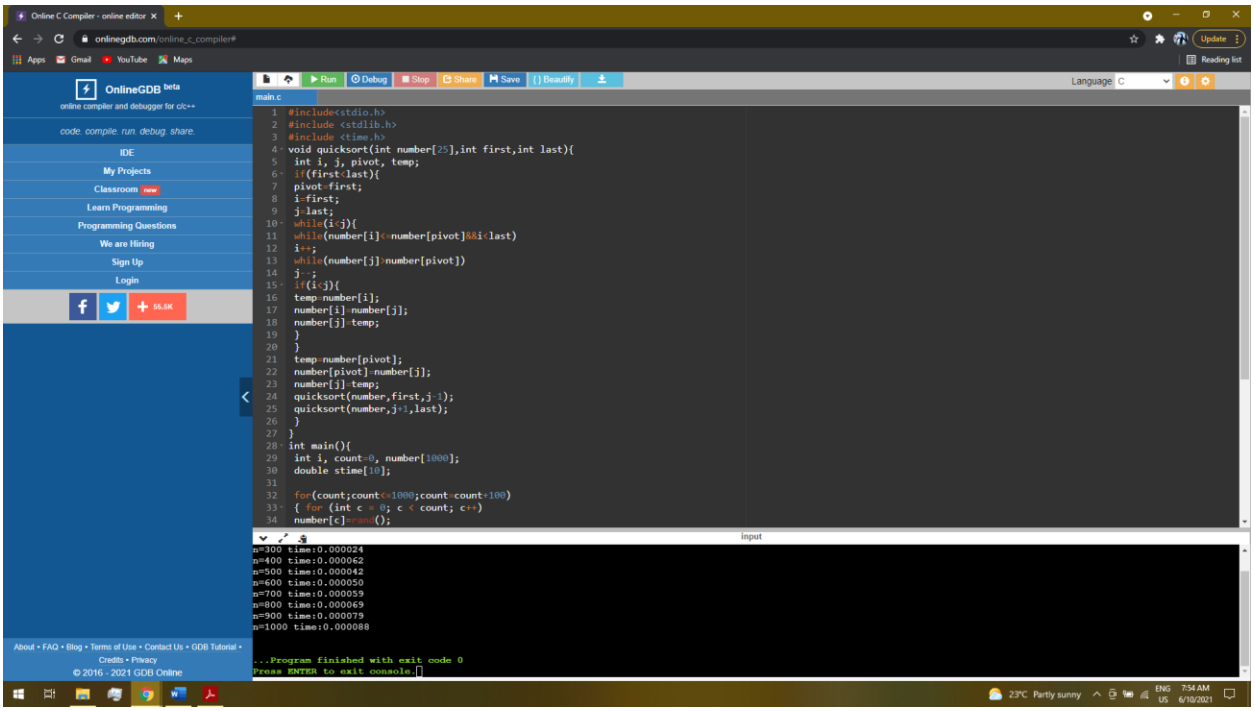
CODE FOR QUICKSORT

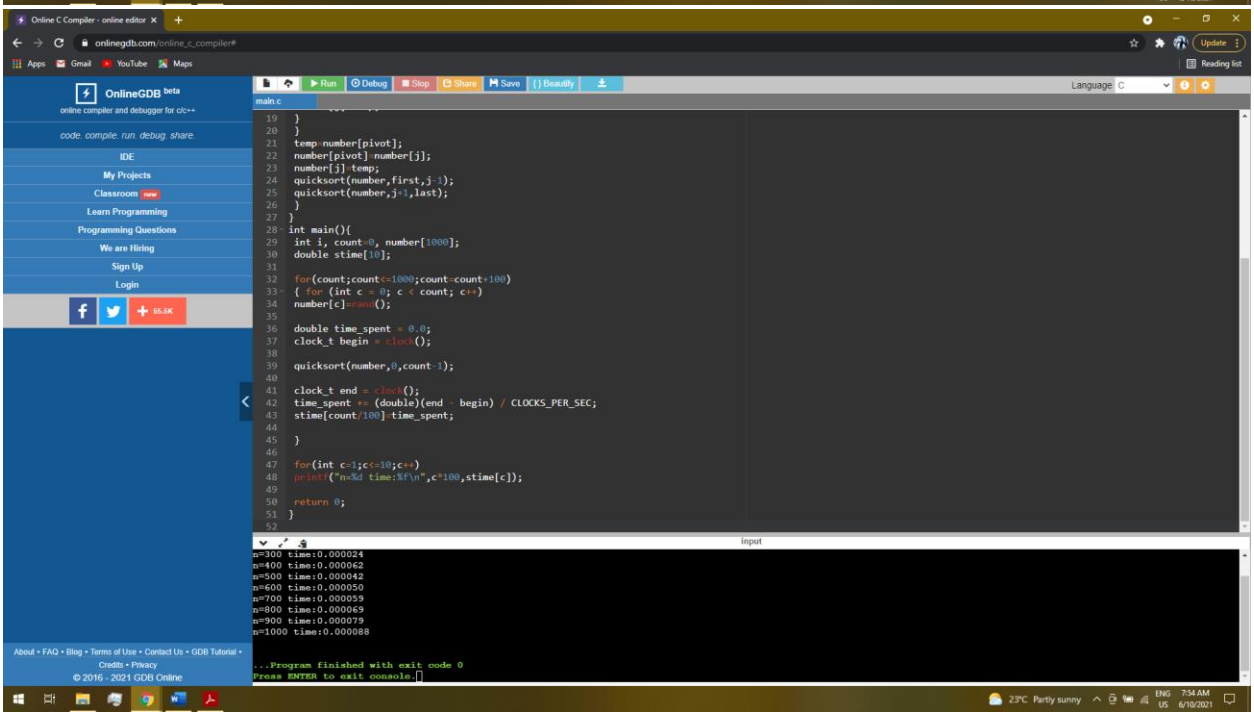
```
#include<stdio.h>

#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) {
```

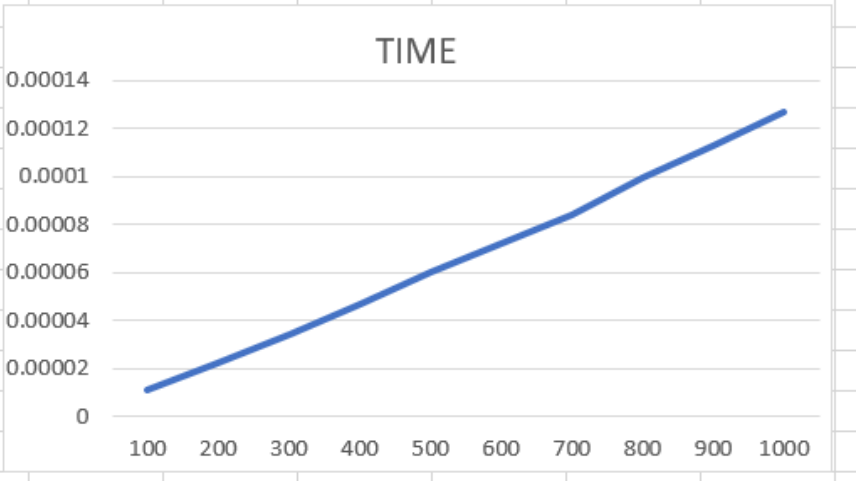
```
 int m = l + (r - l) / 2;
 mergeSort(arr, l, m);
 mergeSort(arr, m + 1, r);
 merge(arr, l, m, r);
 }
}
int main() {
 int arr[1000],count=0;
 double stime[10];
 for(count;count<=1000;count=count+100)
 { for (int c = 0; c < count; c++)
 arr[c]=rand();

 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#
OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
We are Hiring
Sign Up
Login
66.5K
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
#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();
input
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
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
23°C Partly sunny
ENG US
7:54 AM 6/10/2021


Online C Compiler - online editor
onlinegdb.com/online_c_compiler#
OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
We are Hiring
Sign Up
Login
66.5K
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
}
}
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;
}
input
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
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
23°C Partly sunny
ENG US
7:54 AM 6/10/2021

N	TIME
100	0.000011
200	0.000022
300	0.000034
400	0.000047
500	0.00006
600	0.000072
700	0.000084
800	0.000099
900	0.000113
1000	0.000127



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;
}
```