Run Debug Stop Share Save {} Beautify Language C

main.c

```c
#include <stdio.h>
#include<stdlib.h>
#include <time.h>
void swap(int *a, int *b) {
  int t = *a;
  *a = *b;
  *b = t;
}
int partition(int array[], int low, int high) {
  int pivot = array[high];
  int i = (low - 1);
  for (int j = low; j < high; j++) {
    if (array[j] <= pivot) {
      i++;
      swap(&array[i], &array[j]);
    }
  }
  swap(&array[i + 1], &array[high]);
    return (i + 1);
}
void quickSort(int array[], int low, int high) {
  if (low < high) {
    int pi = partition(array, low, high);
    quickSort(array, low, pi - 1);
    quickSort(array, pi + 1, high);
  }
}
void printArray(int array[], int size) {
  for (int i = 0; i < size; ++i) {
```

Connecting...

Search the web and Windows

23:01
26-05-2021

Run Debug Stop Share Save { } Beautify Language C

main.c

```
    int pi = partition(array, low, high);
    quickSort(array, low, pi - 1);
    quickSort(array, pi + 1, high);
  }
}
void printArray(int array[], int size) {
  for (int i = 0; i < size; ++i) {
    printf("%d  ", array[i]);
  }
  printf("\n");
}
int main() {
  int data[100];
  int n;
  while(1){
  printf("Enter no of elements in the array:\n");
  scanf("%d",&n);
  printf("Random %d integers\n",n);
  for(int i=0;i<n;i++)
  data[i]=1+rand()%1000;
   clock_t begin = clock();
    quickSort(data, 0, n - 1);
  printf("Printing the sorted array: \n");
  printArray(data, n);
     clock_t end = clock();
double time_spent = (double)(end - begin) / CLOCKS_PER_SEC;
  printf("\nEXECUTION TIME : %.10fseconds\n\n", time_spent);
  }
}

```

Search the web and Windows

23:02
26-05-2021

input

```
Enter no of elements in the array:
200
Random 200 integers
Printing the sorted array:
12  12  13  20  23  28  30  32  35  43  44  59  60  61  68  70  85  88  92  95  98  98  118  122  124  125  125  136  144  150  168  171  173
 179  194  199  212  220  227  228  229  229  230  236  238  246  271  276  277  281  282  287  302  306  307  314  316  318  325  328  336  3
37  341  351  354  363  365  366  368  369  369  369  371  374  379  380  384  387  394  396  400  404  408  414  422  422  427  429  430  433
  435  435  438  441  442  445  452  457  468  475  482  489  492  493  493  498  501  504  506  527  529  530  531  538  540  541  546  552
556  568  568  571  583  585  587  587  602  619  620  625  650  652  653  676  677  684  690  691  709  710  716  724  730  730  733  737  74
0  744  751  755  757  764  765  765  772  778  783  785  789  794  794  796  797  803  809  815  819  830  842  847  847  857  857  858  859
 860  863  863  866  872  874  887  896  903  915  916  920  922  926  927  928  929  930  933  937  957  966  981  988  997

EXECUTION TIME : 0.0001070000seconds

Enter no of elements in the array:
300
Random 300 integers
Printing the sorted array:
3  5  10  22  28  29  34  37  37  40  43  44  47  50  51  63  72  82  83  85  85  91  96  108  126  128  128  129  132  134  135  140  143  14
5  151  152  153  156  158  160  164  173  178  180  184  189  190  191  191  198  200  203  203  206  207  210  211  216  223  225  229  233
 239  249  250  254  256  256  259  260  270  274  283  286  290  293  293  299  300  300  302  304  304  305  306  312  321  322  325  327  3
34  336  337  337  340  341  341  341  343  344  344  349  349  356  364  368  369  370  373  377  380  380  386  391  397  404  405  413  415
  417  419  423  423  429  430  437  444  445  446  453  461  461  465  466  467  469  471  491  498  499  500  501  501  505  505  506  507
508  523  525  529  530  536  539  539  542  543  543  551  557  568  568  569  570  580  582  588  591  591  600  601  606  607  612  614  61
4  622  623  625  626  627  630  631  638  641  645  648  649  659  660  661  662  668  669  670  673  682  683  685  687  689  693  698  700
 706  709  714  714  722  726  730  731  733  737  744  744  747  747  749  754  755  755  757  764  770  773  774  777  777  785  787  789  7
95  796  798  802  805  809  811  812  814  819  820  820  827  829  830  841  843  851  851  861  869  873  879  882  885  888  891  893  895
  899  900  900  901  903  905  905  909  918  918  921  922  925  927  931  934  937  940  941  945  950  955  959  960  962  965  972  973
974  976  982  985  989  991  994  995  997  997  997

EXECUTION TIME : 0.0001360000seconds
```