|  |  |
| --- | --- |
| **Ch09-02:** *Generic Sorter* | |
| **Line#** | **Code** |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109 | #include <iostream>  #include <string.h>  using namespace std;  typedef int(\*COMPARER)(void\*, void\*);  #define NOI(\_arr) (sizeof(\_arr)/sizeof(\_arr[0]))  #define sort(\_arr,\_cmpr) \_sortEngine(\_arr,NOI(\_arr),sizeof(\_arr[0]),\_cmpr)  #define showStudentRecords(\_students) \_showStudentRecords(\_students,NOI(\_students))  struct STUDENT {  const char\* name;  unsigned char age;  float cgpa;  };  void \_sortEngine(void\* pdata, unsigned n, unsigned sz,  // int (\*comparer)(void\*,void\*)){  COMPARER comparer) {  char\* p;  for (unsigned i = 0; i < (n - 1); i++) {  p = (char\*)pdata;  for (unsigned j = 0; j < (n - 1 - i); j++) {  if (comparer(p, p + sz) > 0) {  for (unsigned z = 0; z < sz; z++) {  char c = p[z];  p[z] = p[z + sz];  p[z + sz] = c;  }  }  p += sz;  }  }  }  int CompareInteger(void\* lhs, void\* rhs) {  return \*((int\*)lhs) - \*((int\*)rhs);  }  int CompareIntegerDesc(void\* lhs, void\* rhs) {  return \*((int\*)rhs) - \*((int\*)lhs);  }  int CompareDouble(void\* lhs, void\* rhs) {  double d1 = \*((double\*)lhs);  double d2 = \*((double\*)rhs);  if (d1 > d2) return 1;  if (d1 < d2) return -1;  return 0;  }  int CompareStudentByAge(void\* lhs, void\* rhs) {  return ((STUDENT\*)lhs)->age - ((STUDENT\*)rhs)->age;  }  int CompareStudentByCGPA(void\* lhs, void\* rhs) {  return (((STUDENT\*)lhs)->cgpa > ((STUDENT\*)rhs)->cgpa) ? 1 : -1;  }  int CompareStudentByName(void\* lhs, void\* rhs) {  return strcmp(((STUDENT\*)lhs)->name, ((STUDENT\*)rhs)->name);  }  void \_showStudentRecords(STUDENT\* ps, unsigned n) {  while (n--) {  cout << "Age:" << (int)ps->age << '\t' <<  "CGPA:" << ps->cgpa << '\t' <<  "Name:" << ps->name << endl;  ps++;  }  cout << endl;  }  int main() {  int idata[] = { 5, 3, 7, 8, 2, 7, 9, 4 };  sort(idata, CompareInteger);  for (int i = 0; i < NOI(idata); i++) cout << idata[i] << '\t';  cout << endl;  sort(idata, CompareIntegerDesc);  for (int i = 0; i < NOI(idata); i++) cout << idata[i] << '\t';  cout << endl;  double ddata[] = { 5.4, 3.2, 7.1, 8.8, 2.4, 7.0, 9.4, 4.1 };  sort(ddata, CompareDouble);  for (int i = 0; i < NOI(ddata); i++) cout << ddata[i] << '\t';  cout << endl;  STUDENT class2020[] = {  {"Fatimah", 19, 3.45F},  {"Zahran", 18, 3.25F},  {"Abu", 20, 3.65F},  {"Zawawi", 20, 3.75F},  {"Nizam", 19, 3.85F},  };  sort(class2020, CompareStudentByAge);  showStudentRecords(class2020);  sort(class2020, CompareStudentByCGPA);  showStudentRecords(class2020);  sort(class2020, CompareStudentByName);  showStudentRecords(class2020);  return 0;  } |

**Your Challenge:**

1) Use Anonymous Function for some given examples

2) Create a struct Circle and an array of Circle. Then use the sort() to sort the circle array by Area