```
Script started on Tue, May 02, 2017  2:18:44 PM
ESC]0;/cygdrive/c/users/essy/desktop/classes/Parallel Processing/prog5BEL

ESC[32messy@essy-HP ESC[33m/cygdrive/c/users/essy/desktop/classes/Parallel Processing/prog5ESC[0m

$ cat prog5.cpp
//// Kenneth Willeford

//// C++ version MPI merge sort

////

//// ....

//// I couldn't get get MPI to run on cygwin, I got it to recognize the header files but it still doesn't see the link libraries.

//// Another attempt I tried was using MS-MPI instead, but I also couldn't get it to run. As such I can't turn in any runtime output.

#include <cstdlib>

#include <iostream>

#include <mpi.h>

using namespace std;



int Compare(const void* a_p, const void* b_p);

void Merge(int local_A[], int local_B[], int local_n);



int main(int argc, char* argv[]){

    int my_rank, comm_sz;

    int n, local_n;

    int *local_A;



    MPI_Init(NULL,NULL);

    MPI_Comm_rank(MPI_COMM_WORLD,&my_rank);

    MPI_Comm_size(MPI_COMM_WORLD,&comm_sz);



    if(my_rank == 0){

        n = atoi(argv[1]);

        if(n % comm_sz != 0){

            cerr << "n should be evenly divisble by " << comm_sz << endl;

            MPI_Finalize();
```

```
62
63              exit(1);
64
65          }
66
67          for(int dest = 1; dest < comm_sz; dest++)
68
69              MPI_Send(&n, 1, MPI_INT, dest, 0, MPI_COMM_WORLD);
70
71      } else {
72
73          MPI_Recv(&n, 1,MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
74
75      }
76
77
78
79      local_n = n/comm_sz; // number of elements for each process
80
81      local_A = new int[n]; // create local list (full size n) in each process, and fill
82
83      srandom(my_rank+1);
84
85      for(int i = 0; i < local_n; i++)
86
87          local_A[i] = random() % 100;
88
89
90
91      cout << "Process_" << my_rank << ", local list:";
92
93      for(int i = 0; i < local_n; i++)
94
95          cout << " " << local_A[i];
96
97      cout << endl;
98
99
100
101     ///// local list sorting in each process
102
103     qsort(local_A, local_n, sizeof(int), Compare);
104
105
106
107     //// reduction (merge) part
108
109     int partner;
110
111     int done = 0;
112
113     int* local_B;
114
115     MPI_Status status;
116
117
118
119     int divisor = 2;
120
121     int core_differ = 1;
122
123     while(done == 0 && divisor <= comm_sz) {
124
125         if(my_rank % divisor == 0){
```

```
126
127         // receiver
128
129             local_B = new int[n];
130
131   //================================================================
132
133   //================================================================
134
135             partner = my_rank + core_differ;
136
137             MPI_Recv(local_B, local_n, MPI_INT, partner, 0, MPI_COMM_WORLD,
                MPI_STATUS_IGNORE);
138
139   //================================================================
140
141   //================================================================
142
143             Merge(local_A, local_B, local_n);
144
145             local_n = 2*local_n; // after merge, double the local_n (size)
146
147             delete[] local_B;
148
149         } else {
150
151         // sender
152
153   //================================================================
154
155   //================================================================
156
157             partner = my_rank - core_differ;
158
159             MPI_Send(local_B,local_n,MPI_INT,partner, 0,MPI_COMM_WORLD);
160
161   //================================================================
162
163   //================================================================
164
165             done = 1; // this (sender) process terminates while loop
166
167         }
168
169         divisor *= 2;
170
171         core_differ *= 2;
172
173      }
174
175
176
177      // Display sorted list
178
179      if(my_rank == 0){
180
181   //================================================================
182
183   //================================================================
184
185         cout << "Sorted List: " << endl;
186
187         for(int i = 0; i < n; i++)
188
```

```
189             cout << local_A
190
191         cout << endl;
192
193 //=====================================================================
194
195 //=====================================================================
196
197     }
198
199
200
201     delete[] local_A;
202
203     MPI_Finalize();
204
205     return 0;
206
207 }
208
209
210
211 int Compare(const void* a_p, const void* b_p){
212
213     int a = *((int*)a_p);
214
215     int b = *((int*)b_p);
216
217
218
219     if (a < b) return -1;
220
221     else if (a == b) return 0;
222
223     else return 1;
224
225 }
226
227
228
229 void Merge(int local_A[], int local_B[], int local_n){
230
231     int ai, bi, ci, i;
232
233     int csize = 2*local_n;
234
235     int* temp_C = new int[csize];
236
237
238
239     ai =0; bi = 0; ci = 0;
240
241     while((ai < local_n) && (bi < local_n)){
242
243         if(local_A[ai] <= local_B[bi]){
244
245             temp_C[ci] = local_A[ai];
246
247             ci++;
248
249             ai++;
250
251         } else {
252
```

```
253             temp_C[ci] = local_B[bi];
254
255             ci++;
256
257             bi++;
258
259         }
260
261     }
262
263
264
265     if(ai >= local_n)
266
267         for(int i = ci; i < csize; i++, bi++)
268
269             temp_C[i] = local_B[bi];
270
271     else if (bi >= local_n)
272
273         for(int i = ci; i < csize; i++, ai++)
274
275             temp_C[i] = local_A[ai];
276
277
278
279     for(int i = 0; i < csize; i++)
280
281         local_A[i] = temp_C[i];
282
283
284
285     delete[] temp_C;
286
287 }ESC]0;/cygdrive/c/users/essy/desktop/classes/Parallel Processing/prog5BEL
288
289 ESC[32messy@essy-HP ESC[33m/cygdrive/c/users/essy/desktop/classes/Parallel
    Processing/prog5ESC[0m
290
291 $ exit
292 exit
293
294 Script done on Tue, May 02, 2017  2:18:49 PM
295
```