Q1

1. 使用command line argument的方式輸入程式參數，ex: ./Q1 90 81 78 95 79 72 85
2. 題目要求將max、min、avg設成global variable以利thread都能共享，另外為了方便將計算max、min、avg函數所需參數包成struct在main thread及worker thread之間傳遞，parameter包含透過command line argument輸入參數個數(size)及輸入參數所組成的陣列(values)。

```c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include<assert.h>

int min = INT_MAX;
int max = INT_MIN;
double avg = 0.0;

typedef struct {
    int size;
    int *values;
}parameters;
```

3. 初始化參數

```c
parameters *data = (parameters*)malloc(sizeof(parameters));
assert(data != NULL);
data->size = argc - 1;
data->values = (int*)calloc(data->size, sizeof(int));
assert(data->values != NULL);
for(int i
    data->values[i] = atoi(argv[i + 1]);
```

int *<unnamed>::values

4. 創建三個worker thread 分別計算max、min、avg，同時傳入parameter，接著使用join() api等待worker thread完成。

```
59      pthread_t minThread, maxThread, avgThread;
60
61      pthread_create(&minThread, NULL, calcMin, data);
62      pthread_create(&maxThread, NULL, calcMax, data);
63      pthread_create(&avgThread, NULL, calcAvg, data);
64
65      pthread_join(minThread, NULL);
66      pthread_join(maxThread, NULL);
67      pthread_join(avgThread, NULL);
68
69      printf("The average value is %lf\n", avg);
70      printf("The minimum value is %d\n", min);
71      printf("The maximum value is %d\n", max);
72
```

5. worker thread 各自的runner function因為max、min、avg皆為global variable所以runner function可以直接使用，另外runner function的parameter在傳入時皆被轉型為void因此在使用前要把他轉回parameter type。

```
20  void *calcMin(void *params){
21      parameters *data = (parameters*) params;
22      int size = data->size;
23      for(int i = 0;i < size;i++)
24          min = min > (data->values)[i] ? (data->values)[i]:min;
25      pthread_exit(0);
26  }
27
28  void *calcMax(void *params){
29      parameters *data = (parameters*) params;
30      int size = data->size;
31      for(int i = 0;i < size;i++)
32          max = max < (data->values)[i] ? (data->values)[i]:max;
33      pthread_exit(0);
```

```
void *calcAvg(void *params){
    parameters *data = (parameters*) params;
    int size = data->size;
    double sum = 0;
    for(int i = 0;i < size;i++)
        sum += (data->values)[i];
    avg = sum / size;
    pthread_exit(0);
}
```

6. 執行結果的截圖，這邊因為avg設為double型態所以不像題目範例是取整數而是連帶後面小數一起印出。

```
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$ gcc Q1.c -o Q1
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$ ./Q1 90 81 78 95 79 72 85
The average value is 82.857143
The minimum value is 72
The maximum value is 95
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$
```

Q2

1. 使用command line argument的方式輸入程式參數，ex: ./Q2 7 12 19 3 18 4 2 6 15
2. 應題目要求需要傳遞參數進worker thread，因此將所需參數包成struct方便在worker thread之間傳遞，parameter包含透過command line argument輸入參數個數(size)、輸入參數所組成的陣列(values)及對應原本陣列起始位置(start)，另外為了區分是哪個worker thread的參數定義各個worker thread所屬parameter struct pointer array的index。

```
 6  #define NumOfThread 3
 7  #define SortParam1 0
 8  #define SortParam2 1
 9  #define MergeParam 2
10
11  typedef struct {
12      int size;
13      int start;
14      int *values;
15  }parameters;
```

3. 初始化兩個sorting thread所需參數，將透過輸入的程式參數平均分成兩半放進代表各個sorting thread的parameter struct中，並同時記錄長度及起始位置。

```c
void initSort(parameters **params, int half, int total, char *argv[]){
    for (int i = 0; i < 2; i++){
        parameters *param = (parameters*)malloc(sizeof(parameters));
        assert(param != NULL);
        if (i % 2){
            param->size = (total - half);
            param->start = half;
        }
        else{
            param->size = half;
            param->start = 0;
        }
        param->values = (int*)calloc(param->size, sizeof(int));
        for (int j = 0; j < param->size; j++){
            param->values[j] = atoi(argv[j + param->start * i + 1]);
        }
        params[i] = param;
    }
}
```

因為總共有3個worker thread(sorting thread1、sorting thread2、merge thread)因此parameters struct array初始化3個。

```c
    int total = argc - 1;
    int half = total / 2;
    parameters *params[NumOfThread];
    initSort(params, half, total, argv);
```

4. 首先創建需要sorting的兩個worker thread，執行sortRunner()function傳入代表該worker thread的parameter struct同時結果回傳代表該worker thread的parameter struct(與傳入時相同)，因此我們驗證只需等待sort thread結束(join())將parameter struct中儲存value印出即可。

```c
    pthread_t sortThread1, sortThread2, mergeThread;

    pthread_create(&sortThread1, NULL, sortRunner, params[SortParam1]);
    pthread_create(&sortThread2, NULL, sortRunner, params[SortParam2]);

    pthread_join(sortThread1, (void **) &params[SortParam1]);
    pthread_join(sortThread2, (void **) &params[SortParam2]);

    for (int i = 0; i < 2; i++){
        printf("Sort Sublist %d: ", i + 1);
        for(int j = 0; j < params[i]->size; j++){
            printf("%d ", params[i]->values[j]);
        }
        printf("\n");
    }
```

sorting algorithm使用merge sort，與Q1相同runner function的parameter在傳入時皆被轉型為void因此在使用前要把他轉回parameter type。

```
40  void *sortRunner(void *params){
41      parameters *data = (parameters*) params;
42      mergeSort(data->values, 0, data->size - 1);
43      pthread_exit(params);
44  }
45
46  void mergeSort(int arr[], int left, int right) {
47      if (left < right) {
48          int mid = left + (right - left) / 2;
49          mergeSort(arr, left, mid);
50          mergeSort(arr, mid + 1, right);
51          merge(arr, left, mid, right);
52      }
53  }
```

```
55  void merge(int arr[], int left, int mid, int right) {
56      int i, j, k;
57      int n1 = mid - left + 1;
58      int n2 = right - mid;
59
60      int L[n1], R[n2];
61
62      for (i = 0; i < n1; i++)
63          L[i] = arr[left + i];
64      for (j = 0; j < n2; j++)
65          R[j] = arr[mid + 1 + j];
66
67      i = 0;
68      j = 0;
69      k = left;
70      while (i < n1 && j < n2) {
71          if (L[i] <= R[j]) {
72              arr[k] = L[i];
73              i++;
74          }
75          else {
76              arr[k] = R[j];
77              j++;
78          }
```

```
79              k++;
80          }
81
82          while (i < n1) {
83              arr[k] = L[i];
84              i++;
85              k++;
86          }
87
88          while (j < n2) {
89              arr[k] = R[j];
90              j++;
91              k++;
92          }
93      }
```

5. 執行完兩個sorting thread，創建merge thread並將兩個各自排好的陣列合併放入代表merge thread的parameter struct，執行mergeRunner()function與sortRunner相同傳入代表該worker thread的parameter struct同時結果回傳代表該worker thread的parameter struct(與傳入時相同)，因此我們驗證只需等待merge thread結束(join())將parameter struct中儲存value印出即可。

```
141     initMerge(params);
142     pthread_create(&mergeThread, NULL, mergeRunner, params[MergeParam]);
143     pthread_join(mergeThread, (void **) &params[MergeParam]);
144
145     printf("Merge Sorted Sublist: ");
146     for(int j = 0; j < params[MergeParam]->size; j++){
147         printf("%d ", params[MergeParam]->values[j]);
148     }
149     printf("\n");
```

```
95  void *mergeRunner(void *params){
96      parameters *data = (parameters*) params;
97      merge(data->values, data->start,
98          (data->start + data->size) / 2 - 1, (data->size) - 1);
99      pthread_exit(params);
100 }
101
102 void initMerge(parameters **data){
103     parameters *param = (parameters*)malloc(sizeof(parameters));
104     param->size = data[SortParam1]->size + data[SortParam2]->size;
105     param->start = 0;
106     param->values = (int*)calloc(param->size, sizeof(int));
107     for (int i = 0; i < 2; i++){
108         for(int j = 0; j < data[i]->size; j++){
109             param->values[data[i]->start + j] = data[i]->values[j];
110         }
111     }
112     data[MergeParam] = param;
113 }
```

6. 最後所有程式執行完畢歸還跟OS要來的記憶體。

```
151     for (int k = 0; k < NumOfThread; k++){
152         free(params[k]->values);
153         free(params[k]);
154         params[k] = NULL;
155     }
156     return 0;
157 }
```

7. 執行結果的截圖，上面表示陣列個數為偶數，下面為奇數。

```
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$ gcc Q2.c -o Q2
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$ ./Q2 7 12 19 3 18 4 2 6 15 8
Sort Sublist 1: 3 7 12 18 19
Sort Sublist 2: 2 4 6 8 15
Merge Sorted Sublist: 2 3 4 6 7 8 12 15 18 19
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$ ./Q2 7 12 19 3 18 4 2 6 15
Sort Sublist 1: 3 7 12 19
Sort Sublist 2: 2 4 6 15 18
Merge Sorted Sublist: 2 3 4 6 7 12 15 18 19
(base) robert@LAPTOP-46RPPSGN:/mnt/c/users/rober/desktop/111-2OS/hw2$
```