Homework #2

TA in charge: Yongwook Lee
E-mail: wooks1998@kaist.ac.kr

I. Goal of this assignment

- ✓ Implementing procedures with recursive calls in MIPS ISA.
- ✓ Writing a MIPS assembly program that implements an algorithm written in a high-level language.

II. What to implement

- ✓ You need to implement the specific quicksort algorithm below:

```
void partition(int A[], int low, int high, int *mid_left_o, int *mid_right_o) {
    int pivot, i;
    int mid_left = low, mid_right = high;

    i = low + (1664525*(unsigned)high + 22695477*(unsigned)low)%(high-low+1);

    pivot = A[i];
    A[i] = A[low];
    A[low] = pivot;

    i = low + 1;
    while (1) {
        while (mid_right >= i && A[mid_right] > pivot) mid_right--;
        while (mid_right >= i && A[i] <= pivot) {
            A[mid_left++] = A[i];
            A[i++] = pivot;
        }
        if (i < mid_right) {
            A[mid_left++] = A[mid_right];
            A[mid_right--] = A[i];
            A[i++] = pivot;
        } else break;
    }

    *mid_left_o = mid_left;
    *mid_right_o = mid_right;
```

```
}

void quicksort(int A[], int low, int high) {
    int mid_left, mid_right;

    if (low < high) {
        partition(A, low, high, &mid_left, &mid_right);
        quicksort(A, low, mid_left - 1);
        quicksort(A, mid_right + 1, high);
    }
}
```

**** Do not implement any other variations of the quicksort algorithm; if your implementation is not equivalent to the given specific algorithm, you will not get any points for this homework.***

✓ You also need to implement the other parts of the program for:

- Receiving input from simulator console in the following format:
  The first line contains an integer **N**, which is the number of integers to be sorted.
  Each of the following lines contains each of the integers to be sorted ($\mathbf{a_1}$, $\mathbf{a_2}$, …, $\mathbf{a_N}$).
  e.g.

```
5
1
4
7
-10
3
```

- Printing output (the result of sorting) to simulator console in the following format:
  Each of the sorted integers in each line (in ascending order; smallest integer in the first line, largest integer in the last line).
  e.g.

```
-10
1
3
4
7
```

III. Constraints

✓ $1 \leq \mathbf{N} \leq 10{,}000$, $-2{,}147{,}483{,}648 \leq \mathbf{a_i} \leq 2{,}147{,}483{,}647$

- ✓ **int** type in the C code is a 32-bit integer type.
- ✓ We will use the same SPIM simulator we used in HW1 to check if your assembly program runs correctly as specified in the given algorithm.
- ✓ *Your program **must be runnable on the simulator**. Otherwise, you will not get any points except for the report. Please check your program before submission.*
- ✓ Your program must terminate after printing the output.

IV. Submission and grading

- ✓ Your submission should include: **(Total 100 pts)**

  A. Source code file of your assembly program [hw2_StudentID.s]
  The source code should contain:

  i. Implementation of the given algorithm in assembly language **(40 pts)**

  ii. Comments explaining the details of your implementation **(20 pts)**

  iii. Code for receiving input integers from simulator console **(5 pts)**

  iv. Code for printing sorted integers to simulator console **(5 pts)**

  B. Brief report [hw2_StudentID.pdf]
  The content of the report should describe:

  i. Stack allocation layout for each procedure **(10 pts)**
  Describe the content of the stack space for each procedure; show the position of each data in the stack space using offset to the stack pointer of each procedure.

  ii. Brief explanation on your implementation **(20 pts)**
  What you have considered for your implementation, implementation issues, and etc.

  There is no specific format for the report. You should submit your code and report in English.

- ✓ **Upload these two files on KLMS**.

V. Due date

- ✓ **End of Oct. 6th (Thu.)**
- ✓ **Late** submission due date: End of Oct. 7th (Fri.)
- ✓ For late submissions, there will be a 50% penalty on your total score.
- ✓ **You cannot submit after the late submission due date**.

VI. Cheating

- ✓ If there are any cheatings in your submission, you will get 0 points.
- ✓ We will do similarity check on the submitted code files to catch plagiarism between students.
- ✓ *Followings will be regarded as cheating*:
  A. Copying other students' simulation result or report
  B. Modifying other students' results and using them as if they were your own
  C. Using other sources without any references
  D. All other sorts of inappropriate behaviors.

VII. Tips & Notes

- ✓ Since SPIM does not simulate delay slot in the default configuration, you can ignore it for this assignment.
- ✓ The simulator may not be able to load your code if it contains any non-English characters.
- ✓ You can use pseudoinstructions supported in SPIM for your convenience.
- ✓ You would need to review how to implement procedure calls with MIPS assembly.
- ✓ **If you have little experience with assembly programming**, this assignment could take much longer than expected. If you think this is your case, we **recommend you to begin this assignment as early as possible**.
- ✓ If you have any questions, please use KLMS Q&A board.
- ✓ You can use Spim(terminal version) when you test your code. You can download from https://sourceforge.net/p/spimsimulator/code/HEAD/tree/.
  Testing command:
  ./spim -ef ../CPU/exceptions.s -file quicksort.s < input.txt > output.txt