All submissions are to be completed through MarkUs. The files submitted must be MIPS assembler files (.asm). Review the course outline for the policy on late submissions. This assignment has two (2) pages.

1. Write a MIPS assembly language program that reads a single line of keyboard input from the user, capitalizes the first letter of each English word in that line, and prints the result as output. You may assume the user will only enter capital letters (A-Z), lowercase letters (a-z), spaces, and a single linefeed \[LF\] character at the end (when the user press the enter key to send input to the program). You may assume words are separated by spaces. Think carefully about corner cases.

For example: if the user entered I like computer science\[LF\], your program should print I Like Computer Science\[LF\].

To receive full marks, your code must run without errors on the noargs simulator available on the UWaterloo student computer science server. There are no subroutines involved in this question. You do not need to follow the register conventions. Format and comment your code to be readable by the markers.

Submit the file a3q1.asm (10 Marks).

2. Write a MIPS assembly language program that contains a subroutine squarediff which calculates the squared difference between two integers passed as arguments.

For example: squarediff(6\$_30\$, 8\$_30\$) = (6\$_30\$ − 8\$_30\$)$^2$ = 4\$_30\$

Call this subroutine on each pair of integers in an input array, and calculate the average of the results.

For example: in the input array [6\$_30\$, 8\$_30\$, 3\$_30\$, 7\$_30\$] there are two pairs (6\$_30\$, 8\$_30\$) and (3\$_30\$, 7\$_30\$). The squarediff results on these pairs are 4\$_30\$ and 16\$_30\$, and the average of those squares (and the final result of the computation) is 10\$_30\$.

If the value of the average is a fraction, round down (use the quotient result from division). You can assume the input array will be of even length, and contain at least two values. You can assume, at the beginning of your program, that the memory address of the start of the array is in register $1$, and the number of items in the array is in register $2$ (this is how the array simulator sets up your program). Place the final result of your computation in register $3$. You may assume all input numbers, intermediate computation results, and final results are valid 32-bit two’s complement integers.

To receive full marks, your code must run without errors on the array simulator available on the UWaterloo student computer science server. Your subroutine must follow the register conventions to receive full marks. Format and comment your code to be readable by the markers.

Submit the file a3q2.asm (10 Marks).
3. Write a MIPS assembly language program that contains a recursive subroutine `sumsuffixmax` which calculates the sum of the largest integer in each suffix array of an input array.

For example: the input array \([6_{10}, 8_{10}, 3_{10}]\) has three suffix arrays \([6_{10}, 8_{10}, 3_{10}]\), \([8_{10}, 3_{10}]\), and \([3_{10}]\). The largest integers in each of these suffixes are \(8_{10}\), \(8_{10}\), and \(3_{10}\). The sum of these largest integers (and the final result of `sumsuffixmax` on this array) is \(19_{10}\).

The `sumsuffixmax` subroutine must take two arguments: the first is the address in memory of the input array, and the second is the length of that array. The `sumsuffixmax` subroutine should call itself recursively on the suffixes of the input array, where each recursive call calculates one maximum. Your program must not contain any code outside of the `sumsuffixmax` subroutine except a short snippet that calls the subroutine with the inputs from the `array` simulator and then ends the program.

An empty array should produce a result of \(0\). You can assume, at the beginning of your program, that the memory address of the start of the input array is in register \(\$1\), and the number of items in the array is in register \(\$2\) (this is how the `array` simulator sets up your program). Leave the final result of your computation in the register it ends up in when `sumsuffixmax` returns according to the register convention. You may assume all input numbers, intermediate computation results, and final results are valid 32-bit two's complement integers.

To receive full marks, your code must run without errors on the `array` simulator available on the UWaterloo student computer science server. Your subroutine must follow the register conventions to receive full marks. Format and comment your code to be readable by the markers.

Submit the file `a3q3.asm` (10 Marks).