HW8.1. SIMD

You have an array of 16384 32-bit integers, and want to double them element-wise (A[i]+=A[i]). You have access to **256**-bit AVX instructions, including addition. How many vector adds must you do to complete this task?

Hint: How many ints can we fit in the 256-bit vector?

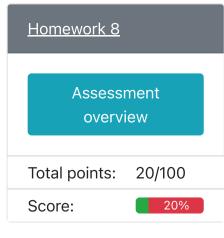
Q1.1: 2048

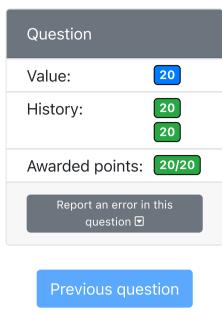
? 100%

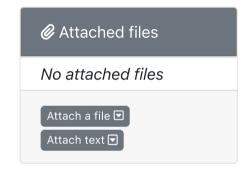
Complete the following code:

```
#include <stdio.h>
#include <stdbool.h>
#include <stdlib.h>
#include <emmintrin.h>
#define N 19
/*Return the first n values in the sequence defined by the recursive
definition A[n] = 7A[n-4]-A[n-8]*/
/*For the initial values of this recursive sequence, see the array used for
n<8*/
int* fib(unsigned int n)
 int* result = malloc(sizeof(int)*n);
 /*Base cases; if we want fewer than 8 inputs, just fill out the array
manually*/
  if(n < 8)
  {
       int vals[8] = \{0,1,1,2,3,5,8,13\};
       for(int i = 0; i < n; i++) result[i]=vals[i];</pre>
       return result:
   _m128i low = _mm_set_epi32(INPUT A); /*See the vals array for the intended
initial values. Make sure you check the Intrinsics guide for the expected
order of the inputs!*/
  _{m128i} high = _{mm_set_epi32(INPUT B)};
  _mm_storeu_si128((__m128i*)result, low);
 _mm_storeu_si128((__m128i*)(result+4), high);
 int i;
 int j = INPUT C;
 for(i = 8; i < j; i+=4)
   //Note that 7x-y is the same as (x<<3)-(x+y)
        D,3),_mm_add_epi32(INPUT E));
       _mm_storeu_si128((__m128i*)(INPUT F), temp);
       low = high;
       high = temp;
 }
 /*Tail case, in case n isn't a multiple of 4.*/
 for(INPUT G) result[i] = 7*result[i-4]-result[i-8];
  return result;
}
int main(int argc, char** argv)
       int* data = fib(N);
       for(int i = 0; i < N; i++)
         printf("%d\n", data[i]);
       free(data);
       return 0;
}
```

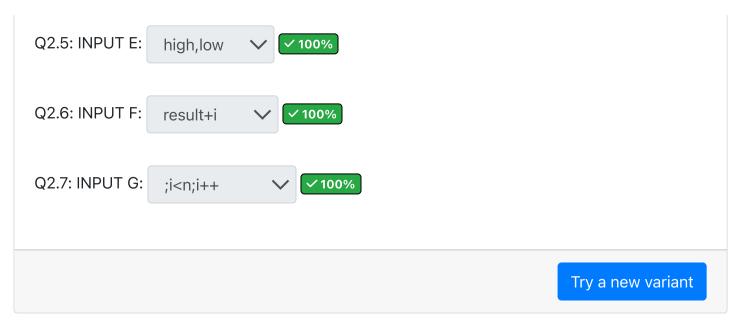


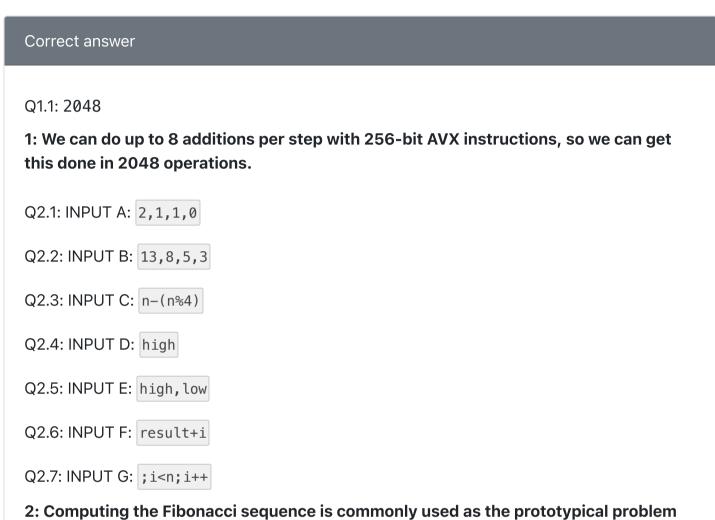






Next question





that is not possible to parallelize, since it requires previous values in order to compute

parallelism on the Fibonacci sequence. Often a parallelizable algorithm looks different

the next value. However, with enough math, it actually is possible to run data-level

from its naive counterpart, so it's often a good idea to take some time to look at the

algorithm itself before starting to parallelize it.

