Here is the link to my work: <https://godbolt.org/z/an9fKK8ef>

As for this homework, I decided to try basic array assignments and standard sort to see the code's performance in the compiler.

From line 2 to line 4, the main function is initialized by copying “rsp” to “rbp” and subtracting “rsp” by 96, where 96 is the size of memory that will be used in the main function. Therefore, rbp points to the beginning or base of the memory address, and rsp points to the top of the stack.

From lines 5 to 6 and lines 14 to 17, the for loop is set. 0 is copied to “rbp - 4”, which is i, and the compiler jumps to line 16 to check whether the value i at “rbp-4” is greater than 10. If so, stop the loop and jump to line 18.

From lines 8 to 13, the codes within the for loop work. First of all, copy the value at “rbp - 4”, which is i, to “eax” for multiplication using “imul”. Then, move the results to “edx” and give “eax” its original value i. After that, move the multiplication results to their correct position, “rbp-48+rax\*4”. Finally, add i by 1 and compare with 10 again to see whether the loop is over.

From lines 18 to 27, 10 values are assigned to the appropriate positions in array b.

From lines 28 to 33, copy the memory address of the first elements in array b to “rax” and move the address to the last element in array b. Then, copy the address of the first element of array b to “rdx”. After that, copy the first element in array b to “rdi” and the last element to “rsi”. Finally, call standard sort functions to sort the array.

From lines 34 to 36, assign the return value of the main function to 0 and return the value. These lines of code ensure the main function returns 0 instead of other values.