Andrew Plum

Professor Bolden

CS 270

10/7/2022

Assignment #1 Results and Report

Results from part 1:

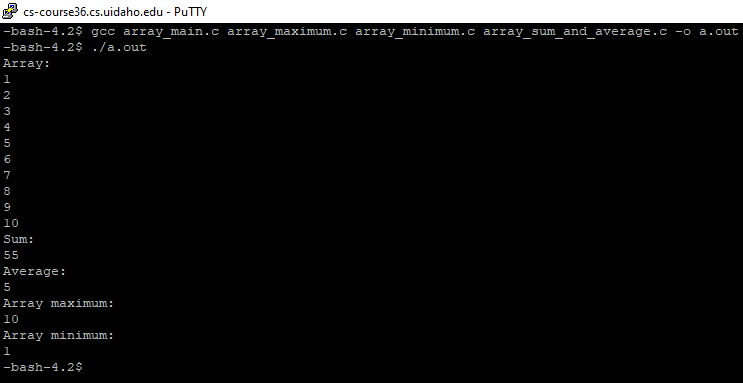
1. Sizes of each function:
   1. Minimum function size: 91 bytes
   2. Maximum function size: 91 bytes
   3. Sum and average function size: 126 bytes
2. Address of where each function is loaded into memory:
   1. Minimum function address: 0x0000000000400664
   2. Maximum function address: 0x0000000000400609
   3. Sum and average function address: 0x00000000004006bf
3. The name and location of the entry point for the entire program:
   1. The name of the entry point for the entire program: \_start
   2. The address of the entry point for the entire program: 0x0000000000400490
4. The locations and names of the functions from stdio that are linked with the file.
   1. Printf
      1. Its name in symbol table: printf@GLIBC\_2.2.5
      2. The location: 0x0000000000601020
5. The locations and sizes of the data sections used by your program.
   1. First .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   2. Second .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   3. Third .data section
      1. Location: 0x000000000060103c
      2. Size: 0x0000000000000000 bytes
   4. Fourth .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   5. Fifth .data section
      1. Location: 0x0000000000601040
      2. Size: 0x0000000000000000 bytes

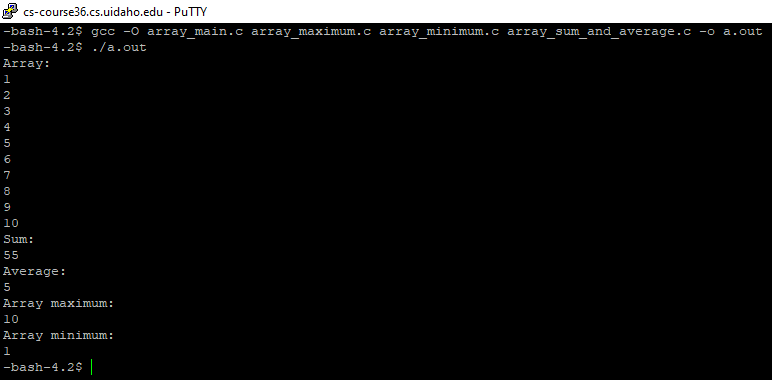
Results from part 2:

1. Sizes of each function:
   1. Minimum function size: 56 bytes
   2. Maximum function size: 56 bytes
   3. Sum and average function size: 80 bytes
2. Address of where each function is loaded into memory:
   1. Minimum function address: 0x000000000040062c
   2. Maximum function address: 0x00000000004005f4
   3. Sum and average function address: 0x0000000000400664
3. The name and location of the entry point for the entire program:
   1. The name of the entry point for the entire program: \_start
   2. The address of the entry point for the entire program: 0x0000000000400490
4. The locations and names of the functions from stdio that are linked with the file.
   1. Printf
      1. Its name in symbol table: printf@GLIBC\_2.2.5
      2. The location: 0x0000000000601020
5. The locations and sizes of the data sections used by your program.
   1. First .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   2. Second .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   3. Third .data section
      1. Location: 0x000000000060103c
      2. Size: 0x0000000000000000 bytes
   4. Fourth .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   5. Fifth .data section
      1. Location: 0x0000000000601040
      2. Size: 0x0000000000000000 bytes

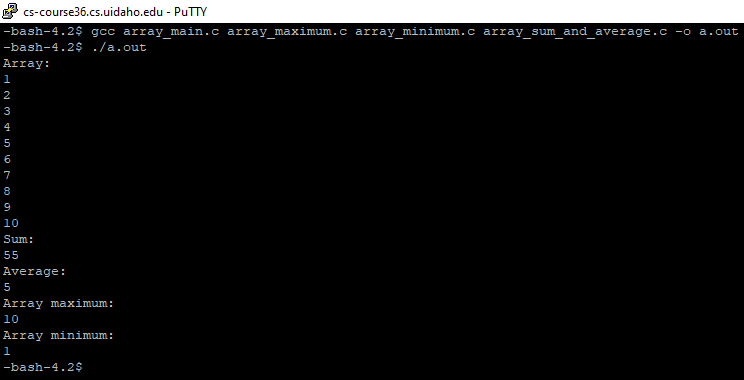
Results from part 3:

1. Sizes of each function:
   1. Minimum function size: 120 bytes
   2. Maximum function size: 120 bytes
   3. Sum and average function size: 128 bytes
2. Address of where each function is loaded into memory:
   1. Minimum function address: 0x0000000000400717
   2. Maximum function address: 0x000000000040069f
   3. Sum and average function address: 0x000000000040078f
3. The name and location of the entry point for the entire program:
   1. The name of the entry point for the entire program: \_start
   2. The address of the entry point for the entire program: 0x0000000000400490
4. The locations and names of the functions from stdio that are linked with the file.
   1. Printf
      1. Its name in symbol table: printf@GLIBC\_2.2.5
      2. The location: 0x0000000000601020
5. The locations and sizes of the data sections used by your program.
   1. First .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   2. Second .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   3. Third .data section
      1. Location: 0x000000000060103c
      2. Size: 0x0000000000000000 bytes
   4. Fourth .data section
      1. Location: 0x0000000000601038
      2. Size: 0x0000000000000000 bytes
   5. Fifth .data section
      1. Location: 0x0000000000601040
      2. Size: 0x0000000000000000 bytes

Screenshots:

Screenshot 1: Files compiled, and executable ran for part 1.

Screenshot 2: Files compiled with optimized option, and executable ran in part 2.

Screenshot 3: Files compiled, and executable ran in part 3.

Conclusion:

Some of the results from the 3 parts were the same and some were different. The results that were different were the size and address of each of the functions. The size of the functions was the tiniest for part 2 which isn’t surprising because the files were compiled using the optimized option. The size of the functions for part 1 were the second largest. And the size of the functions for part 3 were the largest which makes sense because each of functions had to declare the passed in arguments whereas the functions from part 1 and 2 used the already declared external variable. The address of the functions from all 3 parts also varied. All the other results between the 3 parts were the same.

The tools I used to gather the results were readelf, objdump, and nm. I used readelf to gather most of the data from the 3 parts. The only question in all 3 parts I didn’t use it for was when I was finding the locations and sizes of the data sections used by my program; I instead used objdump here. I used nm to verify some of the other results I was finding. Grep was also very useful for searching for the information in the results I needed.

Even though this assignment probably took me longer than it should have, I learned a great deal. I had never used external variables in any of my previous computer science projects, so I had to learn how to use it here. Although I initially ran into an error with the external variables that took a while to solve, I now know how to fix it and won’t make a mistake like that again. I also learned more about readelf, objdump, and nm because I had to use them for this assignment. While researching about them when I was confused, I learned some features of them which may be useful in the future. I also saw firsthand the similarities and differences between when you compile the files with and without the optimize option and when the files use and don’t use external variables. The assignment led me to think about some questions not concerning the assignment that I will try to solve afterwards on my own time.