Systems II Assignment 1:

3a. insertionSort:
Trial 1 = 140.22 seconds
Trial 2 = 136.63 seconds
Cumulative = 276.85

b. quickSort:
Trial 1 = .53
Trial 2 = .47
Cumulative = 1

4a. insertionSort:
Trial 1 = 106.10
Trial 2 = 104.93
Cumulative = 211.03

b. quickSort:
Trial 1 = .52
Trial 2 = .46
Cumulative = .98

5. The -O2 made more of a difference for insertionSort by shedding 65.82 seconds off the cumulative time. The -O2 reduced the speed of the quickSort function by .2 seconds.

6a. Command = objdump -d -j .rodata assign1-0
Result:

```
Disassembly of section .rodata:

0000000000401078 <_IO_stdin_used>:
  401078:       01 00 02 00 00 00 00 00                         ........

0000000000401080 <__dso_handle>:
        ...
  401088:       50 6c 65 61 73 65 20 65 6e 74 65 72 20 25 73 20     Please enter %s
  401098:       00 6e 75 6d 62 65 72 20 6f 66 20 73 74 72 69 6e     .number of strin
  4010a8:       67 73 00 6c 65 6e 67 74 68 20 6f 66 20 65 61 63     gs.length of eac
  4010b8:       68 20 73 74 72 69 6e 67 00 00 00 00 00 00 00 00     h string........
  4010c8:       31 20 66 6f 72 20 69 6e 73 65 72 74 69 6f 6e 20     1 for insertion
  4010d8:       73 6f 72 74 20 6f 72 20 32 20 66 6f 72 20 71 75     sort or 2 for qu
  4010e8:       69 63 6b 73 6f 72 74 00                             icksort.
[gkatsig1@cdmlinux ~]$
```

b. Command = objdump -d -j .text assign1-0
Result:

```
0000000000400b01 <releaseMem>:
  400b01:       55                      push   %rbp
  400b02:       48 89 e5                mov    %rsp,%rbp
  400b05:       48 83 ec 20             sub    $0x20,%rsp
  400b09:       e8 52 fc ff ff          callq  400760 <mcount@plt>
  400b0e:       48 89 7d e8             mov    %rdi,-0x18(%rbp)
  400b12:       89 75 e4                mov    %esi,-0x1c(%rbp)
  400b15:       c7 45 fc 00 00 00 00    movl   $0x0,-0x4(%rbp)
  400b1c:       eb 23                   jmp    400b41 <releaseMem+0x40>
  400b1e:       8b 45 fc                mov    -0x4(%rbp),%eax
  400b21:       48 98                   cltq
  400b23:       48 8d 14 c5 00 00 00    lea    0x0(,%rax,8),%rdx
  400b2a:       00
  400b2b:       48 8b 45 e8             mov    -0x18(%rbp),%rax
  400b2f:       48 01 d0                add    %rdx,%rax
  400b32:       48 8b 00                mov    (%rax),%rax
  400b35:       48 89 c7                mov    %rax,%rdi
  400b38:       e8 73 fb ff ff          callq  4006b0 <free@plt>
  400b3d:       83 45 fc 01             addl   $0x1,-0x4(%rbp)
  400b41:       8b 45 fc                mov    -0x4(%rbp),%eax
  400b44:       3b 45 e4                cmp    -0x1c(%rbp),%eax
  400b47:       7c d5                   jl     400b1e <releaseMem+0x1d>
  400b49:       48 8b 45 e8             mov    -0x18(%rbp),%rax
  400b4d:       48 89 c7                mov    %rax,%rdi
  400b50:       e8 5b fb ff ff          callq  4006b0 <free@plt>
  400b55:       c9                      leaveq
  400b56:       c3                      retq
```

c. Command = objdump -d -j .bss assign1-0
Result:

```
Disassembly of section .bss:

0000000000602090 <stdin@@GLIBC_2.2.5>:
        ...

00000000006020a0 <called.4232>:
  6020a0:       00 00 00 00                                         ....

00000000006020a4 <completed.6345>:
  6020a4:       00 00 00 00                                         ....

00000000006020a8 <strLen>:
        ...
[gkatsig1@cdmlinux ~]$
```

d. The local variable choice in main() cannot be found because assign1-0 is a program. It can be found on the stack at runtime once it is loaded into memory and becomes a process.

7.

One optimization made is that assign1-2 on the right pushed more registers and utilized more registers instead of retrieving from memory like assign1-0 on the left.

```
0000000000400c17 <insertionSort>:
  400c17:       55                      push   %rbp
  400c18:       48 89 e5                mov    %rsp,%rbp
  400c1b:       48 83 ec 20             sub    $0x20,%rsp
  400c1f:       e8 3c fb ff ff          callq  400760 <mcount@plt>
  400c24:       48 89 7d e8             mov    %rdi,-0x18(%rbp)
  400c28:       89 75 e4                mov    %esi,-0x1c(%rbp)
  400c2b:       c7 45 fc 00 00 00 00    movl   $0x0,-0x4(%rbp)
  400c32:       eb 75                   jmp    400ca9 <insertionSort+0x92>
  400c34:       8b 45 fc                mov    -0x4(%rbp),%eax
  400c37:       83 c0 01                add    $0x1,%eax
  400c3a:       89 45 f8                mov    %eax,-0x8(%rbp)
  400c3d:       eb 5e                   jmp    400c9d <insertionSort+0x86>
  400c3f:       8b 05 63 14 20 00       mov    0x201463(%rip),%eax        # 6020a8 <strLen>
  400c45:       48 63 d0                movslq %eax,%rdx
  400c48:       8b 45 f8                mov    -0x8(%rbp),%eax
  400c4b:       48 98                   cltq
  400c4d:       48 8d 0c c5 00 00 00    lea    0x0(,%rax,8),%rcx
  400c54:       00
  400c55:       48 8b 45 e8             mov    -0x18(%rbp),%rax
  400c59:       48 01 c8                add    %rcx,%rax
  400c5c:       48 8b 08                mov    (%rax),%rcx
  400c5f:       8b 45 fc                mov    -0x4(%rbp),%eax
  400c62:       48 98                   cltq
  400c64:       48 8d 34 c5 00 00 00    lea    0x0(,%rax,8),%rsi
  400c6b:       00
  400c6c:       48 8b 45 e8             mov    -0x18(%rbp),%rax
  400c70:       48 01 f0                add    %rsi,%rax
  400c73:       48 8b 00                mov    (%rax),%rax
  400c76:       48 89 ce                mov    %rcx,%rsi
  400c79:       48 89 c7                mov    %rax,%rdi
  400c7c:       e8 4f fa ff ff          callq  4006d0 <strncmp@plt>
  400c81:       85 c0                   test   %eax,%eax
  400c83:       7e 14                   jle    400c99 <insertionSort+0x82>
  400c85:       8b 55 f8                mov    -0x8(%rbp),%edx
  400c88:       8b 4d fc                mov    -0x4(%rbp),%ecx
  400c8b:       48 8b 45 e8             mov    -0x18(%rbp),%rax
  400c8f:       89 ce                   mov    %ecx,%esi
  400c91:       48 89 c7                mov    %rax,%rdi
  400c94:       e8 24 fc ff ff          callq  4008bd <swap>
  400c99:       83 45 f8 01             addl   $0x1,-0x8(%rbp)
  400c9d:       8b 45 f8                mov    -0x8(%rbp),%eax
  400ca0:       3b 45 e4                cmp    -0x1c(%rbp),%eax
  400ca3:       7c 9a                   jl     400c3f <insertionSort+0x28>
  400ca5:       83 45 fc 01             addl   $0x1,-0x4(%rbp)
  400ca9:       8b 45 e4                mov    -0x1c(%rbp),%eax
  400cac:       83 e8 01                sub    $0x1,%eax
  400caf:       3b 45 fc                cmp    -0x4(%rbp),%eax
  400cb2:       7f 80                   jg     400c34 <insertionSort+0x1d>
  400cb4:       c9                      leaveq
  400cb5:       c3                      retq
```

```
0000000000400be0 <insertionSort>:
  400be0:       55                      push   %rbp
  400be1:       48 89 e5                mov    %rsp,%rbp
  400be4:       41 57                   push   %r15
  400be6:       41 56                   push   %r14
  400be8:       41 55                   push   %r13
  400bea:       41 54                   push   %r12
  400bec:       53                      push   %rbx
  400bed:       48 83 ec 18             sub    $0x18,%rsp
  400bf1:       e8 aa fb ff ff          callq  4007a0 <mcount@plt>
  400bf6:       45 31 ff                xor    %r15d,%r15d
  400bf9:       8d 4e ff                lea    -0x1(%rsi),%ecx
  400bfc:       48 89 7d c8             mov    %rdi,-0x38(%rbp)
  400c00:       41 89 f6                mov    %esi,%r14d
  400c03:       4c 8d 6f 08             lea    0x8(%rdi),%r13
  400c07:       89 4d c0                mov    %ecx,-0x40(%rbp)
  400c0a:       44 3b 7d c0             cmp    -0x40(%rbp),%r15d
  400c0e:       7d 58                   jge    400c68 <insertionSort+0x88>
  400c10:       41 8d 47 01             lea    0x1(%r15),%eax
  400c14:       4d 89 ec                mov    %r13,%r12
  400c17:       41 39 c6                cmp    %eax,%r14d
  400c1a:       89 45 c4                mov    %eax,-0x3c(%rbp)
  400c1d:       89 c3                   mov    %eax,%ebx
  400c1f:       7e 39                   jle    400c5a <insertionSort+0x7a>
  400c21:       0f 1f 80 00 00 00 00    nopl   0x0(%rax)
  400c28:       48 63 15 79 14 20 00    movslq 0x201479(%rip),%rdx        # 6020
```

Another optimization made is assign1-2 on the right utilized the bitwise xor, which is much more efficient and quicker.

```
0000000000400f94 <quickSort>:
  400f94:       55                      push   %rbp
  400f95:       48 89 e5                mov    %rsp,%rbp
  400f98:       48 83 ec 10             sub    $0x10,%rsp
  400f9c:       e8 bf f7 ff ff          callq  400760 <mcount@plt>
  400fa1:       48 89 7d f8             mov    %rdi,-0x8(%rbp)
  400fa5:       89 75 f4                mov    %esi,-0xc(%rbp)
  400fa8:       8b 45 f4                mov    -0xc(%rbp),%eax
  400fab:       8d 50 ff                lea    -0x1(%rax),%edx
  400fae:       48 8b 45 f8             mov    -0x8(%rbp),%rax
  400fb2:       be 00 00 00 00          mov    $0x0,%esi
  400fb7:       48 89 c7                mov    %rax,%rdi
  400fba:       e8 46 ff ff ff          callq  400f05 <quickSort_>
  400fbf:       c9                      leaveq
  400fc0:       c3                      retq
  400fc1:       66 2e 0f 1f 84 00 00    nopw   %cs:0x0(%rax,%rax,1)
  400fc8:       00 00 00
  400fcb:       0f 1f 44 00 00          nopl   0x0(%rax,%rax,1)
```

```
0000000000400eb0 <quickSort>:
  400eb0:       55                      push   %rbp
  400eb1:       48 89 e5                mov    %rsp,%rbp
  400eb4:       e8 e7 f8 ff ff          callq  4007a0 <mcount@plt>
  400eb9:       5d                      pop    %rbp
  400eba:       8d 56 ff                lea    -0x1(%rsi),%edx
  400ebd:       31 f6                   xor    %esi,%esi
  400ebf:       e9 7c ff ff ff          jmpq   400e40 <quickSort_>
  400ec4:       66 2e 0f 1f 84 00 00    nopw   %cs:0x0(%rax,%rax,1)
  400ecb:       00 00 00
  400ece:       66 90                   xchg   %ax,%ax
```