# CS 410 Project One Proficiency Test Template

Janera Dobson

Southern New Hampshire University

CS-410-T3249

## Explain the functionality of the blocks of assembly code.

### “main” function”

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| push %rbp  mov %rsp,%rbp  lea 0x0(%rip),%rsi # 0xb <main+11>  lea 0x0(%rip),%rdi # 0x12 <main+18>  callq 0x17 <main+23>  callq 0x1c <main+28> | Push %rbp  Move %rbp in %rsp  Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq <main+23>  Call callq <main+28> |
| mov %eax,0x0(%rip) # 0x22 <main+34>  mov 0x0(%rip),%eax # 0x28 <main+40>  cmp $0x1,%eax  je 0x40 <main+64>  callq 0x40 <main+64> | Move 0(%rip) in %eax  Move 0(%rip) in %eax  Compare %eax by 1  Jump if equal to <main+64>  Call callq <main+64> |
| mov 0x0(%rip),%eax # 0x46 <main+70>  cmp $0x1,%eax  je 0x4d <main+77>  jmp 0x17 <main+23>  callq 0x60 <main+96> | Move %eax in 0(%rip)  Compare %eax in 1  Jump if equal <main+77>  Jump <main+23>  Call callq <main+96> |
| lea 0x0(%rip),%rsi # 0x67 <main+103>  lea 0x0(%rip),%rdi # 0x6e <main+110>  callq 0x73 <main+115>  lea 0x0(%rip),%rsi # 0x7a <main+122>  lea 0x0(%rip),%rdi # 0x81 <main+129>  callq 0x86 <main+134> | Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq <main+115>  Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq <main+134> |
| lea 0x0(%rip),%rsi # 0x8d <main+141>  lea 0x0(%rip),%rdi # 0x94 <main+148>  callq 0x99 <main+153>  lea 0x0(%rip),%rsi # 0xa0 <main+160>  lea 0x0(%rip),%rdi # 0xa7 <main+167>  callq 0xac <main+172> | Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq <main+153>  Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq <main+172> |
| lea 0x0(%rip),%rsi # 0xb3 <main+179>  lea 0x0(%rip),%rdi # 0xba <main+186>  callq 0xbf <main+191>  mov %rax,%rdx  mov 0x0(%rip),%eax # 0xc8 <main+200>  mov %eax,%esi  mov %rdx,%rdi  callq 0xd2 <main+210> | Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq <main+192>  Move %rdx in %rax  Move %eax in 0(%rip)  Move %esi in %eax  Move %rdi in %rdi  Call callq <main+210> |
| mov %rax,%rdx  mov 0x0(%rip),%rax # 0xdc <main+220>  mov %rax,%rsi  mov %rdx,%rdi  callq 0xe7 <main+231> | Move %rdx in %rax  Move %rax in 0(%rip)  Move %rsi in %rax  Move %rdi in %rdx  Call callq <main+231> |
| mov 0x0(%rip),%eax # 0xed <main+237>  cmp $0x1,%eax  jne 0xf9 <main+249>  callq 0xf7 <main+247> | Move %eax in 0(%rip)  Compare %eax by 1  Conditional jump that follows previous test  Call callq <main+247> |
| jmp 0x109 <main+265>  mov 0x0(%rip),%eax # 0xff <main+255>  cmp $0x2,%eax  jne 0x109 <main+265>  callq 0x109 <main+265> | Jump to <main+265>  Move %eax in 0(%rip)  Compare %eax by 2  Conditional jump that follows previous test  Call callq <main+265> |
| mov 0x0(%rip),%eax # 0x10f <main+271>  cmp $0x3,%eax  je 0x119 <main+281>  jmpq 0x4d <main+77> | Move %eax in 0(%rip)  Compare %eax by 3  Jump if equal to <main+281>  Indirect jump to <main+77> |
| mov $0x0,%eaxp pop %rbp  0x000000000000 retq | Removes %rbp  return |

### ChangeCustomerChoice function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| mov 0x0(%rip),%eax # 0x483 <\_Z20ChangeCustomerChoicev+86>  cmp $0x1,%eax  jne 0x496 <\_Z20ChangeCustomerChoicev+105>  mov 0x0(%rip),%eax # 0x48e <\_Z20ChangeCustomerChoicev+97>  mov %eax,0x0(%rip) # 0x494 <\_Z20ChangeCustomerChoicev+103>  jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | Move %eax in 0(%rip)  Compare %eax by 1  Conditional jump that follows previous test  Move %ex in 0(%rip)  Move 0(%rip) in %eax  Jump |
| mov 0x0(%rip),%eax # 0x49c <\_Z20ChangeCustomerChoicev+111>  cmp $0x2,%eax  jne 0x4af <\_Z20ChangeCustomerChoicev+130>  mov 0x0(%rip),%eax # 0x4a7 <\_Z20ChangeCustomerChoicev+122>  mov %eax,0x0(%rip) # 0x4ad <\_Z20ChangeCustomerChoicev+128>  jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | Move %eax in 0(%rip)  Compare %eax by 2  Conditional jump that follows previous test  Move %eax in 0(%rip)  Move 0(%rip) in %eax  Jump |
| mov 0x0(%rip),%eax # 0x4b5 <\_Z20ChangeCustomerChoicev+136>  cmp $0x3,%eax  jne 0x4c8 <\_Z20ChangeCustomerChoicev+155>  mov 0x0(%rip),%eax # 0x4c0 <\_Z20ChangeCustomerChoicev+147>  mov %eax,0x0(%rip) # 0x4c6 <\_Z20ChangeCustomerChoicev+153>  jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | Move %eax in 0(%rip)  Compare %eax by 3  Conditional jump that follows previous test  Move %eax in 0(%rip)  Move 0(%rip) in %eax  Jump |
| mov 0x0(%rip),%eax # 0x4ce <\_Z20ChangeCustomerChoicev+161>  cmp $0x4,%eax  jne 0x4e1 <\_Z20ChangeCustomerChoicev+180>  mov 0x0(%rip),%eax # 0x4d9 <\_Z20ChangeCustomerChoicev+172>  mov %eax,0x0(%rip) # 0x4df <\_Z20ChangeCustomerChoicev+178>  jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203>  mov 0x0(%rip),%eax # 0x4e7 <\_Z20ChangeCustomerChoicev+186> | Move %eax in 0(%rip)  Compare %eax by 4  Conditional jump that follows previous test  Move %eax in 0(%rip)  Move 0(%rip) in %eax  Jump  Move %eax in 0(%rip) |
| mov 0x0(%rip),%eax # 0x4e7 <\_Z20ChangeCustomerChoicev+186>  cmp $0x5,%eax  jne 0x4f8 <\_Z20ChangeCustomerChoicev+203>  mov 0x0(%rip),%eax # 0x4f2 <\_Z20ChangeCustomerChoicev+197>  mov %eax,0x0(%rip) # 0x4f8 <\_Z20ChangeCustomerChoicev+203>  nop  pop %rbp  retq | Move %eax in 0(%rip)  Compare %eax by 5  Conditional jump that follows previous test  Move %eax in 0(%rip)  Move 0(%rip) in %eax  Jump  Removes %rbp  return |

### CheckUserPermissonAccess Function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| lea -0x40(%rbp),%rax  mov %rax,%rdi  callq 0x1cd <\_Z25CheckUserPermissionAccessv+173>  mov %eax,-0x44(%rbp)  cmpl $0x0,-0x44(%rbp)  jne 0x1dd <\_Z25CheckUserPermissionAccessv+189>  mov $0x1,%ebx  jmp 0x1e2 <\_Z25CheckUserPermissionAccessv+194>  mov $0x2,%ebx  lea -0x40(%rbp),%rax  mov %rax,%rdi  callq 0x1ee <\_Z25CheckUserPermissionAccessv+206>  mov %ebx,%eax | Load %rax in -40(%rax)  Move %rdi in %rax  Call callq  Move -44(%rbp) in %eax  Compare Logic -44(%rbp) by 0  Conditional jump that follows previous test  Move %ebx by 1  Jump  Move %ebx by 2  Load %rax in -40(%rbp)  Move %rdi in %rax  Call callq  Move %eax in %ebx |
| mov -0x18(%rbp),%rcx  xor %fs:0x28,%rcx  je 0x23a <\_Z25CheckUserPermissionAccessv+282>  jmp 0x235 <\_Z25CheckUserPermissionAccessv+277>  mov %rax,%rbx  lea -0x45(%rbp),%rax  mov %rax,%rdi  callq 0x210 <\_Z25CheckUserPermissionAccessv+240> | Move %rcx in -18(%rbp)  Bitwise operation %rcx in hidden base  Jump if equal  Jump  Move %rbx in %rax  Load %rax in -45(%rbp)  Move %rdi in %rax  Call callq |
| mov %rbx,%rax  mov %rax,%rdi  callq 0x21b <\_Z25CheckUserPermissionAccessv+251>  mov %rax,%rbx  lea -0x40(%rbp),%rax  mov %rax,%rdi  callq 0x22a <\_Z25CheckUserPermissionAccessv+266>] | Move %rax in %rbx  Move %rdi in %rax  Call callq  Move %rbx in %rax  Load %rax in -40(%rbp)  Move %rdi in %rax  Call callq |
| mov %rbx,%rax  mov %rax,%rdi  callq 0x235 <\_Z25CheckUserPermissionAccessv+277>  callq 0x23a <\_Z25CheckUserPermissionAccessv+282>  add $0x48,%rsp  pop %rbx  pop %rbp  retq | Move %rax in %rbx  Move %rdi in %rax  Call callq  Call callq  Add %rsp in allocated space of 0\*48  Remove %rbx  Remove %rbp  return |

### DisplayInfo Function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| mov %rax,%rdx  mov 0x0(%rip),%rax # 0x262 <\_Z11DisplayInfov+33>  mov %rax,%rsi  mov %rdx,%rdi  callq 0x26d <\_Z11DisplayInfov+44>  lea 0x0(%rip),%rsi # 0x274 <\_Z11DisplayInfov+51>  lea 0x0(%rip),%rdi # 0x27b <\_Z11DisplayInfov+58>  callq 0x280 <\_Z11DisplayInfov+63> | Move %rdx in %rax  Move %rax in 0(%rip)  Move %rsi in %rax  Move %rdi in %rdx  Call callq  Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Call callq |
| lea 0x0(%rip),%rsi # 0x287 <\_Z11DisplayInfov+70>  mov %rax,%rdi  callq 0x28f <\_Z11DisplayInfov+78>  lea 0x0(%rip),%rsi # 0x296 <\_Z11DisplayInfov+85>  mov %rax,%rdi  callq 0x29e <\_Z11DisplayInfov+93>  mov %rax,%rdx  mov 0x0(%rip),%eax # 0x2a7 <\_Z11DisplayInfov+102>  mov %eax,%esi  mov %rdx,%rdi  callq 0x2b1 <\_Z11DisplayInfov+112>  mov %rax,%rdx  mov 0x0(%rip),%rax # 0x2bb <\_Z11DisplayInfov+122>  mov %rax,%rsi  mov %rdx,%rdi  callq 0x2c6 <\_Z11DisplayInfov+133> | Load %rsi in 0(%rip)  Move %rdi in %rax  Call callq  Load %rsi in 0(%rip)  Move %rdi in %rax  Call callq  Move %rdx in %rax  Move %eax in 0(%rip)  Move %esi in %eax  Move %rdi in %rdx  Call callq  Move %rdx in %rax  Move %rax in 0(%rip)  Move %rsi in %rax  Move %rdi in %rdx  Call callq |
| lea 0x0(%rip),%rsi # 0x2cd <\_Z11DisplayInfov+140>  lea 0x0(%rip),%rdi # 0x2d4 <\_Z11DisplayInfov+147>  callq 0x2d9 <\_Z11DisplayInfov+152>  lea 0x0(%rip),%rsi # 0x2e0 <\_Z11DisplayInfov+159>  mov %rax,%rdi  callq 0x2e8 <\_Z11DisplayInfov+167>  lea 0x0(%rip),%rsi # 0x2ef <\_Z11DisplayInfov+174>  mov %rax,%rdi  callq 0x2f7 <\_Z11DisplayInfov+182>  mov %rax,%rdx  mov 0x0(%rip),%eax # 0x300 <\_Z11DisplayInfov+191>  mov %eax,%esi  mov %rdx,%rdi  callq 0x30a <\_Z11DisplayInfov+201>  mov %rax,%rdx  mov 0x0(%rip),%rax # 0x314 <\_Z11DisplayInfov+211>  mov %rax,%rsi  mov %rdx,%rdi  callq 0x31f <\_Z11DisplayInfov+222> | Load %rsi in 0(%rrip)  Load %rdi in 0(%rip)  Call callq  Load %rsi in 0(%rip)  Move %rdi in %rax  Call callq  Load %rsi in 0(%rip)  Move %rdi in %rax  Call callq  Move %rdx in %rax  Move %eax in 0(%rip)  Move %esi in %eax  Move %rdi iin %rdx  Call callq  Move %rdx in %rax  Move %rax in 0(%rip)  Move %rsi in %rax  Move %rdi in %rdx  Call callq |
| mov %rax,%rdx  mov 0x0(%rip),%rax # 0x3c6 <\_Z11DisplayInfov+389>  mov %rax,%rsi  mov %rdx,%rdi  callq 0x3d1 <\_Z11DisplayInfov+400>  lea 0x0(%rip),%rsi # 0x3d8 <\_Z11DisplayInfov+407>  lea 0x0(%rip),%rdi # 0x3df <\_Z11DisplayInfov+414>  callq 0x3e4 <\_Z11DisplayInfov+419>  lea 0x0(%rip),%rsi # 0x3eb <\_Z11DisplayInfov+426>  mov %rax,%rdi  callq 0x3f3 <\_Z11DisplayInfov+434>  lea 0x0(%rip),%rsi # 0x3fa <\_Z11DisplayInfov+441>  mov %rax,%rdi  callq 0x402 <\_Z11DisplayInfov+449> | Move %rdx in %rax  Move %rax in 0(%rrip)  Move %rsi in %rax  Move %rdi in %rdx  Call callq  Load %rsi in 0(%rip)  Load %rdi in 0(%rip)  Ccall callq  Load %rsi in 0(%rip)  Move %rdi in %rax  Call callq  Load %rsi in 0(%rip)  Move %rdi in %rax  Call callq |
| mov %rax,%rdx  mov 0x0(%rip),%eax # 0x40b <\_Z11DisplayInfov+458>  mov %rax,%rdx  mov 0x0(%rip),%rax # 0x41f <\_Z11DisplayInfov+478>  mov %rax,%rsi  mov %rdx,%rdi  callq 0x42a <\_Z11DisplayInfov+489>  nop  pop %rbp  retq | Move %rdx in %rax  Move %eax in 0(%rip)  Move %rdx in %rax  Move %rax in 0(%rip)  Move %rsi in %rax  Move %rdi in %rdx  Call callq  No operation  Remove %rbp  return |

Non-debugging symbols:

0x0000000000000000 main

0x0000000000000120 CheckUserPermissionAccess()

0x0000000000000241 DisplayInfo()

0x000000000000042d ChangeCustomerChoice()

0x00000000000004fb \_\_static\_initialization\_and\_destruction\_0(int, int)

0x0000000000000544 \_GLOBAL\_\_sub\_I\_username

(gdb) disassemble main

Dump of assembler code for function main:

0x0000000000000000 <+0>: push %rbp

0x0000000000000001 <+1>: mov %rsp,%rbp

0x0000000000000004 <+4>: lea 0x0(%rip),%rsi # 0xb <main+11>

0x000000000000000b <+11>: lea 0x0(%rip),%rdi # 0x12 <main+18>

0x0000000000000012 <+18>: callq 0x17 <main+23>

0x0000000000000017 <+23>: callq 0x1c <main+28>

0x000000000000001c <+28>: mov %eax,0x0(%rip) # 0x22 <main+34>

0x0000000000000022 <+34>: mov 0x0(%rip),%eax # 0x28 <main+40>

0x0000000000000028 <+40>: cmp $0x1,%eax

0x000000000000002b <+43>: je 0x40 <main+64>

0x000000000000002d <+45>: lea 0x0(%rip),%rsi # 0x34 <main+52>

0x0000000000000034 <+52>: lea 0x0(%rip),%rdi # 0x3b <main+59>

0x000000000000003b <+59>: callq 0x40 <main+64>

0x0000000000000040 <+64>: mov 0x0(%rip),%eax # 0x46 <main+70>

0x0000000000000046 <+70>: cmp $0x1,%eax

0x0000000000000049 <+73>: je 0x4d <main+77>

0x000000000000004b <+75>: jmp 0x17 <main+23>

0x000000000000004d <+77>: lea 0x0(%rip),%rsi # 0x54 <main+84>

0x0000000000000054 <+84>: lea 0x0(%rip),%rdi # 0x5b <main+91>

0x000000000000005b <+91>: callq 0x60 <main+96>

0x0000000000000060 <+96>: lea 0x0(%rip),%rsi # 0x67 <main+103>

0x0000000000000067 <+103>: lea 0x0(%rip),%rdi # 0x6e <main+110>

0x000000000000006e <+110>: callq 0x73 <main+115>

0x0000000000000073 <+115>: lea 0x0(%rip),%rsi # 0x7a <main+122>

0x000000000000007a <+122>: lea 0x0(%rip),%rdi # 0x81 <main+129>

0x0000000000000081 <+129>: callq 0x86 <main+134>

0x0000000000000086 <+134>: lea 0x0(%rip),%rsi # 0x8d <main+141>

0x000000000000008d <+141>: lea 0x0(%rip),%rdi # 0x94 <main+148>

0x0000000000000094 <+148>: callq 0x99 <main+153>

0x0000000000000099 <+153>: lea 0x0(%rip),%rsi # 0xa0 <main+160>

0x00000000000000a0 <+160>: lea 0x0(%rip),%rdi # 0xa7 <main+167>

0x00000000000000a7 <+167>: callq 0xac <main+172>

0x00000000000000ac <+172>: lea 0x0(%rip),%rsi # 0xb3 <main+179>

0x00000000000000b3 <+179>: lea 0x0(%rip),%rdi # 0xba <main+186>

0x00000000000000ba <+186>: callq 0xbf <main+191>

0x00000000000000bf <+191>: mov %rax,%rdx

0x00000000000000c2 <+194>: mov 0x0(%rip),%eax # 0xc8 <main+200>

0x00000000000000c8 <+200>: mov %eax,%esi

0x00000000000000ca <+202>: mov %rdx,%rdi

0x00000000000000cd <+205>: callq 0xd2 <main+210>

0x00000000000000d2 <+210>: mov %rax,%rdx

0x00000000000000d5 <+213>: mov 0x0(%rip),%rax # 0xdc <main+220>

0x00000000000000dc <+220>: mov %rax,%rsi

0x00000000000000df <+223>: mov %rdx,%rdi

0x00000000000000e2 <+226>: callq 0xe7 <main+231>

0x00000000000000e7 <+231>: mov 0x0(%rip),%eax # 0xed <main+237>

0x00000000000000ed <+237>: cmp $0x1,%eax

0x00000000000000f0 <+240>: jne 0xf9 <main+249>

---Type <return> to continue, or q <return> to quit---return

0x00000000000000f2 <+242>: callq 0xf7 <main+247>

0x00000000000000f7 <+247>: jmp 0x109 <main+265>

0x00000000000000f9 <+249>: mov 0x0(%rip),%eax # 0xff <main+255>

0x00000000000000ff <+255>: cmp $0x2,%eax

0x0000000000000102 <+258>: jne 0x109 <main+265>

0x0000000000000104 <+260>: callq 0x109 <main+265>

0x0000000000000109 <+265>: mov 0x0(%rip),%eax # 0x10f <main+271>

0x000000000000010f <+271>: cmp $0x3,%eax

0x0000000000000112 <+274>: je 0x119 <main+281>

0x0000000000000114 <+276>: jmpq 0x4d <main+77>

0x0000000000000119 <+281>: mov $0x0,%eax

0x000000000000011e <+286>: pop %rbp

0x000000000000011f <+287>: retq

End of assembler dump.

c

Dump of assembler code for function \_Z25CheckUserPermissionAccessv:

0x0000000000000120 <+0>: push %rbp

0x0000000000000121 <+1>: mov %rsp,%rbp

0x0000000000000124 <+4>: push %rbx

0x0000000000000125 <+5>: sub $0x48,%rsp

0x0000000000000129 <+9>: mov %fs:0x28,%rax

0x0000000000000132 <+18>: mov %rax,-0x18(%rbp)

0x0000000000000136 <+22>: xor %eax,%eax

0x0000000000000138 <+24>: lea -0x45(%rbp),%rax

0x000000000000013c <+28>: mov %rax,%rdi

0x000000000000013f <+31>: callq 0x144 <\_Z25CheckUserPermissionAccessv+36>

0x0000000000000144 <+36>: lea -0x45(%rbp),%rdx

0x0000000000000148 <+40>: lea -0x40(%rbp),%rax

0x000000000000014c <+44>: lea 0x0(%rip),%rsi # 0x153 <\_Z25CheckUserPermissionAccessv+51>

0x0000000000000153 <+51>: mov %rax,%rdi

0x0000000000000156 <+54>: callq 0x15b <\_Z25CheckUserPermissionAccessv+59>

0x000000000000015b <+59>: lea -0x45(%rbp),%rax

0x000000000000015f <+63>: mov %rax,%rdi

0x0000000000000162 <+66>: callq 0x167 <\_Z25CheckUserPermissionAccessv+71>

0x0000000000000167 <+71>: movl $0x0,-0x44(%rbp)

0x000000000000016e <+78>: lea 0x0(%rip),%rsi # 0x175 <\_Z25CheckUserPermissionAccessv+85>

0x0000000000000175 <+85>: lea 0x0(%rip),%rdi # 0x17c <\_Z25CheckUserPermissionAccessv+92>

0x000000000000017c <+92>: callq 0x181 <\_Z25CheckUserPermissionAccessv+97>

0x0000000000000181 <+97>: lea 0x0(%rip),%rsi # 0x188 <\_Z25CheckUserPermissionAccessv+104>

0x0000000000000188 <+104>: lea 0x0(%rip),%rdi # 0x18f <\_Z25CheckUserPermissionAccessv+111>

0x000000000000018f <+111>: callq 0x194 <\_Z25CheckUserPermissionAccessv+116>

0x0000000000000194 <+116>: lea 0x0(%rip),%rsi # 0x19b <\_Z25CheckUserPermissionAccessv+123>

0x000000000000019b <+123>: lea 0x0(%rip),%rdi # 0x1a2 <\_Z25CheckUserPermissionAccessv+130>

0x00000000000001a2 <+130>: callq 0x1a7 <\_Z25CheckUserPermissionAccessv+135>

0x00000000000001a7 <+135>: lea -0x40(%rbp),%rax

0x00000000000001ab <+139>: mov %rax,%rsi

0x00000000000001ae <+142>: lea 0x0(%rip),%rdi # 0x1b5 <\_Z25CheckUserPermissionAccessv+149>

0x00000000000001b5 <+149>: callq 0x1ba <\_Z25CheckUserPermissionAccessv+154>

0x00000000000001ba <+154>: lea -0x40(%rbp),%rax

0x00000000000001be <+158>: lea 0x0(%rip),%rsi # 0x1c5 <\_Z25CheckUserPermissionAccessv+165>

0x00000000000001c5 <+165>: mov %rax,%rdi

0x00000000000001c8 <+168>: callq 0x1cd <\_Z25CheckUserPermissionAccessv+173>

0x00000000000001cd <+173>: mov %eax,-0x44(%rbp)

0x00000000000001d0 <+176>: cmpl $0x0,-0x44(%rbp)

0x00000000000001d4 <+180>: jne 0x1dd <\_Z25CheckUserPermissionAccessv+189>

0x00000000000001d6 <+182>: mov $0x1,%ebx

0x00000000000001db <+187>: jmp 0x1e2 <\_Z25CheckUserPermissionAccessv+194>

0x00000000000001dd <+189>: mov $0x2,%ebx

0x00000000000001e2 <+194>: lea -0x40(%rbp),%rax

0x00000000000001e6 <+198>: mov %rax,%rdi

0x00000000000001e9 <+201>: callq 0x1ee <\_Z25CheckUserPermissionAccessv+206>

0x00000000000001ee <+206>: mov %ebx,%eax

0x00000000000001f0 <+208>: mov -0x18(%rbp),%rcx

0x00000000000001f4 <+212>: xor %fs:0x28,%rcx

---Type <return> to continue, or q <return> to quit---return

0x00000000000001fd <+221>: je 0x23a <\_Z25CheckUserPermissionAccessv+282>

0x00000000000001ff <+223>: jmp 0x235 <\_Z25CheckUserPermissionAccessv+277>

0x0000000000000201 <+225>: mov %rax,%rbx

0x0000000000000204 <+228>: lea -0x45(%rbp),%rax

0x0000000000000208 <+232>: mov %rax,%rdi

0x000000000000020b <+235>: callq 0x210 <\_Z25CheckUserPermissionAccessv+240>

0x0000000000000210 <+240>: mov %rbx,%rax

0x0000000000000213 <+243>: mov %rax,%rdi

0x0000000000000216 <+246>: callq 0x21b <\_Z25CheckUserPermissionAccessv+251>

0x000000000000021b <+251>: mov %rax,%rbx

0x000000000000021e <+254>: lea -0x40(%rbp),%rax

0x0000000000000222 <+258>: mov %rax,%rdi

0x0000000000000225 <+261>: callq 0x22a <\_Z25CheckUserPermissionAccessv+266>

0x000000000000022a <+266>: mov %rbx,%rax

0x000000000000022d <+269>: mov %rax,%rdi

0x0000000000000230 <+272>: callq 0x235 <\_Z25CheckUserPermissionAccessv+277>

0x0000000000000235 <+277>: callq 0x23a <\_Z25CheckUserPermissionAccessv+282>

0x000000000000023a <+282>: add $0x48,%rsp

0x000000000000023e <+286>: pop %rbx

0x000000000000023f <+287>: pop %rbp

0x0000000000000240 <+288>: retq

End of assembler dump.

(gdb) disassemble DisplayInfo

Dump of assembler code for function \_Z11DisplayInfov:

0x0000000000000241 <+0>: push %rbp

0x0000000000000242 <+1>: mov %rsp,%rbp

0x0000000000000245 <+4>: lea 0x0(%rip),%rsi # 0x24c <\_Z11DisplayInfov+11>

0x000000000000024c <+11>: lea 0x0(%rip),%rdi # 0x253 <\_Z11DisplayInfov+18>

0x0000000000000253 <+18>: callq 0x258 <\_Z11DisplayInfov+23>

0x0000000000000258 <+23>: mov %rax,%rdx

0x000000000000025b <+26>: mov 0x0(%rip),%rax # 0x262 <\_Z11DisplayInfov+33>

0x0000000000000262 <+33>: mov %rax,%rsi

0x0000000000000265 <+36>: mov %rdx,%rdi

0x0000000000000268 <+39>: callq 0x26d <\_Z11DisplayInfov+44>

0x000000000000026d <+44>: lea 0x0(%rip),%rsi # 0x274 <\_Z11DisplayInfov+51>

0x0000000000000274 <+51>: lea 0x0(%rip),%rdi # 0x27b <\_Z11DisplayInfov+58>

0x000000000000027b <+58>: callq 0x280 <\_Z11DisplayInfov+63>

0x0000000000000280 <+63>: lea 0x0(%rip),%rsi # 0x287 <\_Z11DisplayInfov+70>

0x0000000000000287 <+70>: mov %rax,%rdi

0x000000000000028a <+73>: callq 0x28f <\_Z11DisplayInfov+78>

0x000000000000028f <+78>: lea 0x0(%rip),%rsi # 0x296 <\_Z11DisplayInfov+85>

0x0000000000000296 <+85>: mov %rax,%rdi

0x0000000000000299 <+88>: callq 0x29e <\_Z11DisplayInfov+93>

0x000000000000029e <+93>: mov %rax,%rdx

0x00000000000002a1 <+96>: mov 0x0(%rip),%eax # 0x2a7 <\_Z11DisplayInfov+102>

0x00000000000002a7 <+102>: mov %eax,%esi

0x00000000000002a9 <+104>: mov %rdx,%rdi

0x00000000000002ac <+107>: callq 0x2b1 <\_Z11DisplayInfov+112>

0x00000000000002b1 <+112>: mov %rax,%rdx

0x00000000000002b4 <+115>: mov 0x0(%rip),%rax # 0x2bb <\_Z11DisplayInfov+122>

0x00000000000002bb <+122>: mov %rax,%rsi

0x00000000000002be <+125>: mov %rdx,%rdi

0x00000000000002c1 <+128>: callq 0x2c6 <\_Z11DisplayInfov+133>

0x00000000000002c6 <+133>: lea 0x0(%rip),%rsi # 0x2cd <\_Z11DisplayInfov+140>

0x00000000000002cd <+140>: lea 0x0(%rip),%rdi # 0x2d4 <\_Z11DisplayInfov+147>

0x00000000000002d4 <+147>: callq 0x2d9 <\_Z11DisplayInfov+152>

0x00000000000002d9 <+152>: lea 0x0(%rip),%rsi # 0x2e0 <\_Z11DisplayInfov+159>

0x00000000000002e0 <+159>: mov %rax,%rdi

0x00000000000002e3 <+162>: callq 0x2e8 <\_Z11DisplayInfov+167>

0x00000000000002e8 <+167>: lea 0x0(%rip),%rsi # 0x2ef <\_Z11DisplayInfov+174>

0x00000000000002ef <+174>: mov %rax,%rdi

0x00000000000002f2 <+177>: callq 0x2f7 <\_Z11DisplayInfov+182>

0x00000000000002f7 <+182>: mov %rax,%rdx

0x00000000000002fa <+185>: mov 0x0(%rip),%eax # 0x300 <\_Z11DisplayInfov+191>

0x0000000000000300 <+191>: mov %eax,%esi

0x0000000000000302 <+193>: mov %rdx,%rdi

0x0000000000000305 <+196>: callq 0x30a <\_Z11DisplayInfov+201>

0x000000000000030a <+201>: mov %rax,%rdx

0x000000000000030d <+204>: mov 0x0(%rip),%rax # 0x314 <\_Z11DisplayInfov+211>

0x0000000000000314 <+211>: mov %rax,%rsi

0x0000000000000317 <+214>: mov %rdx,%rdi

0x000000000000031a <+217>: callq 0x31f <\_Z11DisplayInfov+222>

---Type <return> to continue, or q <return> to quit---return

0x000000000000031f <+222>: lea 0x0(%rip),%rsi # 0x326 <\_Z11DisplayInfov+229>

0x0000000000000326 <+229>: lea 0x0(%rip),%rdi # 0x32d <\_Z11DisplayInfov+236>

0x000000000000032d <+236>: callq 0x332 <\_Z11DisplayInfov+241>

0x0000000000000332 <+241>: lea 0x0(%rip),%rsi # 0x339 <\_Z11DisplayInfov+248>

0x0000000000000339 <+248>: mov %rax,%rdi

0x000000000000033c <+251>: callq 0x341 <\_Z11DisplayInfov+256>

0x0000000000000341 <+256>: lea 0x0(%rip),%rsi # 0x348 <\_Z11DisplayInfov+263>

0x0000000000000348 <+263>: mov %rax,%rdi

0x000000000000034b <+266>: callq 0x350 <\_Z11DisplayInfov+271>

0x0000000000000350 <+271>: mov %rax,%rdx

0x0000000000000353 <+274>: mov 0x0(%rip),%eax # 0x359 <\_Z11DisplayInfov+280>

0x0000000000000359 <+280>: mov %eax,%esi

0x000000000000035b <+282>: mov %rdx,%rdi

0x000000000000035e <+285>: callq 0x363 <\_Z11DisplayInfov+290>

0x0000000000000363 <+290>: mov %rax,%rdx

0x0000000000000366 <+293>: mov 0x0(%rip),%rax # 0x36d <\_Z11DisplayInfov+300>

0x000000000000036d <+300>: mov %rax,%rsi

0x0000000000000370 <+303>: mov %rdx,%rdi

0x0000000000000373 <+306>: callq 0x378 <\_Z11DisplayInfov+311>

0x0000000000000378 <+311>: lea 0x0(%rip),%rsi # 0x37f <\_Z11DisplayInfov+318>

0x000000000000037f <+318>: lea 0x0(%rip),%rdi # 0x386 <\_Z11DisplayInfov+325>

0x0000000000000386 <+325>: callq 0x38b <\_Z11DisplayInfov+330>

0x000000000000038b <+330>: lea 0x0(%rip),%rsi # 0x392 <\_Z11DisplayInfov+337>

0x0000000000000392 <+337>: mov %rax,%rdi

0x0000000000000395 <+340>: callq 0x39a <\_Z11DisplayInfov+345>

0x000000000000039a <+345>: lea 0x0(%rip),%rsi # 0x3a1 <\_Z11DisplayInfov+352>

0x00000000000003a1 <+352>: mov %rax,%rdi

0x00000000000003a4 <+355>: callq 0x3a9 <\_Z11DisplayInfov+360>

0x00000000000003a9 <+360>: mov %rax,%rdx

0x00000000000003ac <+363>: mov 0x0(%rip),%eax # 0x3b2 <\_Z11DisplayInfov+369>

0x00000000000003b2 <+369>: mov %eax,%esi

0x00000000000003b4 <+371>: mov %rdx,%rdi

0x00000000000003b7 <+374>: callq 0x3bc <\_Z11DisplayInfov+379>

0x00000000000003bc <+379>: mov %rax,%rdx

0x00000000000003bf <+382>: mov 0x0(%rip),%rax # 0x3c6 <\_Z11DisplayInfov+389>

0x00000000000003c6 <+389>: mov %rax,%rsi

0x00000000000003c9 <+392>: mov %rdx,%rdi

0x00000000000003cc <+395>: callq 0x3d1 <\_Z11DisplayInfov+400>

0x00000000000003d1 <+400>: lea 0x0(%rip),%rsi # 0x3d8 <\_Z11DisplayInfov+407>

0x00000000000003d8 <+407>: lea 0x0(%rip),%rdi # 0x3df <\_Z11DisplayInfov+414>

0x00000000000003df <+414>: callq 0x3e4 <\_Z11DisplayInfov+419>

0x00000000000003e4 <+419>: lea 0x0(%rip),%rsi # 0x3eb <\_Z11DisplayInfov+426>

0x00000000000003eb <+426>: mov %rax,%rdi

0x00000000000003ee <+429>: callq 0x3f3 <\_Z11DisplayInfov+434>

0x00000000000003f3 <+434>: lea 0x0(%rip),%rsi # 0x3fa <\_Z11DisplayInfov+441>

0x00000000000003fa <+441>: mov %rax,%rdi

0x00000000000003fd <+444>: callq 0x402 <\_Z11DisplayInfov+449>

0x0000000000000402 <+449>: mov %rax,%rdx

0x0000000000000405 <+452>: mov 0x0(%rip),%eax # 0x40b <\_Z11DisplayInfov+458>

---Type <return> to continue, or q <return> to quit---return

0x000000000000040b <+458>: mov %eax,%esi

0x000000000000040d <+460>: mov %rdx,%rdi

0x0000000000000410 <+463>: callq 0x415 <\_Z11DisplayInfov+468>

0x0000000000000415 <+468>: mov %rax,%rdx

0x0000000000000418 <+471>: mov 0x0(%rip),%rax # 0x41f <\_Z11DisplayInfov+478>

0x000000000000041f <+478>: mov %rax,%rsi

0x0000000000000422 <+481>: mov %rdx,%rdi

0x0000000000000425 <+484>: callq 0x42a <\_Z11DisplayInfov+489>

0x000000000000042a <+489>: nop

0x000000000000042b <+490>: pop %rbp

0x000000000000042c <+491>: retq

End of assembler dump.

(gdb) disassemble ChangeCustomerChoice

Dump of assembler code for function \_Z20ChangeCustomerChoicev:

0x000000000000042d <+0>: push %rbp

0x000000000000042e <+1>: mov %rsp,%rbp

0x0000000000000431 <+4>: lea 0x0(%rip),%rsi # 0x438 <\_Z20ChangeCustomerChoicev+11>

0x0000000000000438 <+11>: lea 0x0(%rip),%rdi # 0x43f <\_Z20ChangeCustomerChoicev+18>

0x000000000000043f <+18>: callq 0x444 <\_Z20ChangeCustomerChoicev+23>

0x0000000000000444 <+23>: lea 0x0(%rip),%rsi # 0x44b <\_Z20ChangeCustomerChoicev+30>

0x000000000000044b <+30>: lea 0x0(%rip),%rdi # 0x452 <\_Z20ChangeCustomerChoicev+37>

0x0000000000000452 <+37>: callq 0x457 <\_Z20ChangeCustomerChoicev+42>

0x0000000000000457 <+42>: lea 0x0(%rip),%rsi # 0x45e <\_Z20ChangeCustomerChoicev+49>

0x000000000000045e <+49>: lea 0x0(%rip),%rdi # 0x465 <\_Z20ChangeCustomerChoicev+56>

0x0000000000000465 <+56>: callq 0x46a <\_Z20ChangeCustomerChoicev+61>

0x000000000000046a <+61>: lea 0x0(%rip),%rsi # 0x471 <\_Z20ChangeCustomerChoicev+68>

0x0000000000000471 <+68>: lea 0x0(%rip),%rdi # 0x478 <\_Z20ChangeCustomerChoicev+75>

0x0000000000000478 <+75>: callq 0x47d <\_Z20ChangeCustomerChoicev+80>

0x000000000000047d <+80>: mov 0x0(%rip),%eax # 0x483 <\_Z20ChangeCustomerChoicev+86>

0x0000000000000483 <+86>: cmp $0x1,%eax

0x0000000000000486 <+89>: jne 0x496 <\_Z20ChangeCustomerChoicev+105>

0x0000000000000488 <+91>: mov 0x0(%rip),%eax # 0x48e <\_Z20ChangeCustomerChoicev+97>

0x000000000000048e <+97>: mov %eax,0x0(%rip) # 0x494 <\_Z20ChangeCustomerChoicev+103>

0x0000000000000494 <+103>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203>

0x0000000000000496 <+105>: mov 0x0(%rip),%eax # 0x49c <\_Z20ChangeCustomerChoicev+111>

0x000000000000049c <+111>: cmp $0x2,%eax

0x000000000000049f <+114>: jne 0x4af <\_Z20ChangeCustomerChoicev+130>

0x00000000000004a1 <+116>: mov 0x0(%rip),%eax # 0x4a7 <\_Z20ChangeCustomerChoicev+122>

0x00000000000004a7 <+122>: mov %eax,0x0(%rip) # 0x4ad <\_Z20ChangeCustomerChoicev+128>

0x00000000000004ad <+128>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203>

0x00000000000004af <+130>: mov 0x0(%rip),%eax # 0x4b5 <\_Z20ChangeCustomerChoicev+136>

0x00000000000004b5 <+136>: cmp $0x3,%eax

0x00000000000004b8 <+139>: jne 0x4c8 <\_Z20ChangeCustomerChoicev+155>

0x00000000000004ba <+141>: mov 0x0(%rip),%eax # 0x4c0 <\_Z20ChangeCustomerChoicev+147>

0x00000000000004c0 <+147>: mov %eax,0x0(%rip) # 0x4c6 <\_Z20ChangeCustomerChoicev+153>

0x00000000000004c6 <+153>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203>

0x00000000000004c8 <+155>: mov 0x0(%rip),%eax # 0x4ce <\_Z20ChangeCustomerChoicev+161>

0x00000000000004ce <+161>: cmp $0x4,%eax

0x00000000000004d1 <+164>: jne 0x4e1 <\_Z20ChangeCustomerChoicev+180>

0x00000000000004d3 <+166>: mov 0x0(%rip),%eax # 0x4d9 <\_Z20ChangeCustomerChoicev+172>

0x00000000000004d9 <+172>: mov %eax,0x0(%rip) # 0x4df <\_Z20ChangeCustomerChoicev+178>

0x00000000000004df <+178>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203>

0x00000000000004e1 <+180>: mov 0x0(%rip),%eax # 0x4e7 <\_Z20ChangeCustomerChoicev+186>

0x00000000000004e7 <+186>: cmp $0x5,%eax

0x00000000000004ea <+189>: jne 0x4f8 <\_Z20ChangeCustomerChoicev+203>

0x00000000000004ec <+191>: mov 0x0(%rip),%eax # 0x4f2 <\_Z20ChangeCustomerChoicev+197>

0x00000000000004f2 <+197>: mov %eax,0x0(%rip) # 0x4f8 <\_Z20ChangeCustomerChoicev+203>

0x00000000000004f8 <+203>: nop

0x00000000000004f9 <+204>: pop %rbp

0x00000000000004fa <+205>: retq

End of assembler dump.

(gdb)