# CS 410 Project One Proficiency Test Template

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

### “main” function”

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x0(%rip),%rsi # 0xb <main+11>  <+11>: lea 0x0(%rip),%rdi # 0x12 <main+18>  <+18>: callq 0x17 <main+23> | Moves %rbp onto the stack.  Moves the value in %rsp into %rbp  Loads %rip into %rsi  Loads %rip ino %rdi  Calls the function located at 0x17 (“Hello! Welcome to our Investment Company”) |
| <+23>: callq 0x1c <main+28>  <+28>: mov %eax,0x0(%rip) # 0x22 <main+34>  <+34>: mov 0x0(%rip),%eax # 0x28 <main+40> | Forms a While loop that calls the CheckUserPermissions. It returns a bool variable and stores it in 0x0(%rip) |
| <+40>: cmp $0x1,%eax  <+43>: je 0x40 <main+64> | If CheckUserPermissions returns with a true bool variable it then jumps to 0x40 |
| <+45>: lea 0x0(%rip),%rsi # 0x34 <main+52>  <+52>: lea 0x0(%rip),%rdi # 0x3b <main+59>  <+59>: callq 0x40 <main+64> | If CheckUserPermissions returns with a false bool variable, it outputs “Invalid Password. Try Again.” |
| <+64>: mov 0x0(%rip),%eax # 0x46 <main+70>  <+70>: cmp $0x1,%eax  <+73>: je 0x4d <main+77>  <+75>: jmp 0x17 <main+23> | Loop that loops while if user permission is not valid and loops to the beginning, if it is valid exits the loop |
| <+77>: lea 0x0(%rip),%rsi # 0x54 <main+84>  <+84>: lea 0x0(%rip),%rdi # 0x5b <main+91>  <+91>: callq 0x60 <main+96>  <+96>: lea 0x0(%rip),%rsi # 0x67 <main+103>  <+103>: lea 0x0(%rip),%rdi # 0x6e <main+110>  <+110>: callq 0x73 <main+115>  <+115>: lea 0x0(%rip),%rsi # 0x7a <main+122>  <+122>: lea 0x0(%rip),%rdi # 0x81 <main+129>  <+129>: callq 0x86 <main+134>  <+134>: lea 0x0(%rip),%rsi # 0x8d <main+141>  <+141>: lea 0x0(%rip),%rdi # 0x94 <main+148>  <+148>: callq 0x99 <main+153> | While loop begins  Output of menu options  What would you like to do?  DISPLAY the client list (enter 1)  CHANGE a client’s choice (enter 2)  EXIT the program (enter 3) |
| <+153>: lea 0x0(%rip),%rsi # 0xa0 <main+160>  <+160>: lea 0x0(%rip),%rdi # 0xa7 <main+167>  <+167>: callq 0xac <main+172> | Reads the user’s input and stores it in 0x0(%rip) |
| <+172>: lea 0x0(%rip),%rsi # 0xb3 <main+179>  <+179>: lea 0x0(%rip),%rdi # 0xba <main+186>  <+186>: callq 0xbf <main+191>  <+191>: mov %rax,%rdx  <+194>: mov 0x0(%rip),%eax # 0xc8 <main+200>  <+200>: mov %eax,%esi  <+202>: mov %rdx,%rdi  <+205>: callq 0xd2 <main+210>  <+210>: mov %rax,%rdx  <+213>: mov 0x0(%rip),%rax # 0xdc <main+220>  <+220>: mov %rax,%rsi  <+223>: mov %rdx,%rdi  <+226>: callq 0xe7 <main+231> | Outputs the message that will validate the user’s input  “You Chose” user input variable |
| <+249>: mov 0x0(%rip),%eax # 0xff <main+255>  <+255>: cmp $0x2,%eax  <+258>: jne 0x109 <main+265>  <+260>: callq 0x109 <main+265>  <+265>: mov 0x0(%rip),%eax # 0x10f <main+271>  <+271>: cmp $0x3,%eax  <+274>: je 0x119 <main+281>  <+276>: jmpq 0x4d <main+77> | If user input is 1 then call function DisplayInfo  If user input is 2 then call function ChangeCustomerChoice  if user input is 3 then exits the loop and program  if the user enters a number that isn’t a choice returns to the start of the loop. |
| <+281>: mov $0x0,%eax  <+286>: pop %rbp  <+287>: retq |  |

### ChangeCustomerChoice function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x0(%rip),%rsi # 0x438 <\_Z20ChangeCustomerChoicev+11>  <+11>: lea 0x0(%rip),%rdi # 0x43f <\_Z20ChangeCustomerChoicev+18>  <+18>: callq 0x444 <\_Z20ChangeCustomerChoicev+23> | Prints out the string “Enter the number of the client that you wish to change\n” |
| <+23>: lea 0x0(%rip),%rsi # 0x44b <\_Z20ChangeCustomerChoicev+30>  <+30>: lea 0x0(%rip),%rdi # 0x452 <\_Z20ChangeCustomerChoicev+37>  <+37>: callq 0x457 <\_Z20ChangeCustomerChoicev+42> | Reads the user’s input and stores it in 0x0(%rip) or int changeChoice |
| <+42>: lea 0x0(%rip),%rsi # 0x45e <\_Z20ChangeCustomerChoicev+49>  <+49>: lea 0x0(%rip),%rdi # 0x465 <\_Z20ChangeCustomerChoicev+56>  <+56>: callq 0x46a <\_Z20ChangeCustomerChoicev+61> | Prints out the message “Please enter the client’s new service choice (1 = Brokerage, 2 = Retirement)\n” |
| <+61>: lea 0x0(%rip),%rsi # 0x471 <\_Z20ChangeCustomerChoicev+68>  <+68>: lea 0x0(%rip),%rdi # 0x478 <\_Z20ChangeCustomerChoicev+75>  <+75>: callq 0x47d <\_Z20ChangeCustomerChoicev+80> | Reads the user’s input and stores it in 0x0(%rip) or int newService |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x0(%rip),%rsi # 0x438 <\_Z20ChangeCustomerChoicev+11>  <+11>: lea 0x0(%rip),%rdi # 0x43f <\_Z20ChangeCustomerChoicev+18>  <+18>: callq 0x444 <\_Z20ChangeCustomerChoicev+23> | Prints out the string “Enter the number of the client that you wish to change\n” |
| <+80>: mov 0x0(%rip),%eax # 0x483 <\_Z20ChangeCustomerChoicev+86>  <+86>: cmp $0x1,%eax  <+89>: jne 0x496 <\_Z20ChangeCustomerChoicev+105>  <+91>: mov 0x0(%rip),%eax # 0x48e <\_Z20ChangeCustomerChoicev+97>  <+97>: mov %eax,0x0(%rip) # 0x494 <\_Z20ChangeCustomerChoicev+103>  <+103>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | If the variable changeChoice is equal to 1 then set the global variable num1 to int newService |
| <+105>: mov 0x0(%rip),%eax # 0x49c <\_Z20ChangeCustomerChoicev+111>  <+111>: cmp $0x2,%eax  <+114>: jne 0x4af <\_Z20ChangeCustomerChoicev+130>  <+116>: mov 0x0(%rip),%eax # 0x4a7 <\_Z20ChangeCustomerChoicev+122>  <+122>: mov %eax,0x0(%rip) # 0x4ad <\_Z20ChangeCustomerChoicev+128>  <+128>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | If the variable changeChoice is equal to 2 then set the global varibale num2 to int newService |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x0(%rip),%rsi # 0x438 <\_Z20ChangeCustomerChoicev+11>  <+11>: lea 0x0(%rip),%rdi # 0x43f <\_Z20ChangeCustomerChoicev+18>  <+18>: callq 0x444 <\_Z20ChangeCustomerChoicev+23> | Prints out the string “Enter the number of the client that you wish to change\n” |
| <+130>: mov 0x0(%rip),%eax # 0x4b5 <\_Z20ChangeCustomerChoicev+136>  <+136>: cmp $0x3,%eax  <+139>: jne 0x4c8 <\_Z20ChangeCustomerChoicev+155>  <+141>: mov 0x0(%rip),%eax # 0x4c0 <\_Z20ChangeCustomerChoicev+147>  <+147>: mov %eax,0x0(%rip) # 0x4c6 <\_Z20ChangeCustomerChoicev+153>  <+153>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | If the variable changeChoice is equal to 3 then set the global variable num3 to int newService |
| <+155>: mov 0x0(%rip),%eax # 0x4ce <\_Z20ChangeCustomerChoicev+161>  <+161>: cmp $0x4,%eax  <+164>: jne 0x4e1 <\_Z20ChangeCustomerChoicev+180>  <+166>: mov 0x0(%rip),%eax # 0x4d9 <\_Z20ChangeCustomerChoicev+172>  <+172>: mov %eax,0x0(%rip) # 0x4df <\_Z20ChangeCustomerChoicev+178>  <+178>: jmp 0x4f8 <\_Z20ChangeCustomerChoicev+203> | If the variable changeChoice is equal to 4 then set the global variable num4 to int newService |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x0(%rip),%rsi # 0x438 <\_Z20ChangeCustomerChoicev+11>  <+11>: lea 0x0(%rip),%rdi # 0x43f <\_Z20ChangeCustomerChoicev+18>  <+18>: callq 0x444 <\_Z20ChangeCustomerChoicev+23> | Prints out the string “Enter the number of the client that you wish to change\n” |
| <+180>: mov 0x0(%rip),%eax # 0x4e7 <\_Z20ChangeCustomerChoicev+186>  <+186>: cmp $0x5,%eax  <+189>: jne 0x4f8 <\_Z20ChangeCustomerChoicev+203>  <+191>: mov 0x0(%rip),%eax # 0x4f2 <\_Z20ChangeCustomerChoicev+197>  <+197>: mov %eax,0x0(%rip) # 0x4f8 <\_Z20ChangeCustomerChoicev+203> | If the variable changeChoice is equal to 5 then set global variable num5 to int newService |

### CheckUserPermissonAccess Function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: push %rbx  <+5>: sub $0x48,%rsp  <+9>: mov %fs:0x28,%rax  <+18>: mov %rax,-0x18(%rbp)  <+22>: xor %eax,%eax  <+24>: lea -0x45(%rbp),%rax  <+28>: mov %rax,%rdi  <+31>: callq 0x144 <\_Z25CheckUserPermissionAccessv+36> | Pushes the value of %rbp onto the stack  Moves %rsp onto %rbp  Pushes the value of %rbx onto the stack  Subtracts 72 bytes from the %rsp to allocate space |
| <+36>: lea -0x45(%rbp),%rdx  <+40>: lea -0x40(%rbp),%rax  <+44>: lea 0x0(%rip),%rsi # 0x153 <\_Z25CheckUserPermissionAccessv+51>  <+51>: mov %rax,%rdi  <+54>: callq 0x15b <\_Z25CheckUserPermissionAccessv+59>  <+59>: lea -0x45(%rbp),%rax  <+63>: mov %rax,%rdi  <+66>: callq 0x167 <\_Z25CheckUserPermissionAccessv+71>  <+71>: movl $0x0,-0x44(%rbp)  <+78>: lea 0x0(%rip),%rsi # 0x175 <\_Z25CheckUserPermissionAccessv+85>  <+85>: lea 0x0(%rip),%rdi # 0x17c <\_Z25CheckUserPermissionAccessv+92>  <+92>: callq 0x181 <\_Z25CheckUserPermissionAccessv+97> | Prints out “Enter your username: \n” to prompt user to input their username |
| <+97>: lea 0x0(%rip),%rsi # 0x188 <\_Z25CheckUserPermissionAccessv+104>  <+104>: lea 0x0(%rip),%rdi # 0x18f <\_Z25CheckUserPermissionAccessv+111>  <+111>: callq 0x194 <\_Z25CheckUserPermissionAccessv+116> | Reads the user’s input to 0x0(%rip) or String username |
| <+116>: lea 0x0(%rip),%rsi # 0x19b <\_Z25CheckUserPermissionAccessv+123>  <+123>: lea 0x0(%rip),%rdi # 0x1a2 <\_Z25CheckUserPermissionAccessv+130>  <+130>: callq 0x1a7 <\_Z25CheckUserPermissionAccessv+135> | Prints out “Enter your password: \n” to prompt user to enter their password |
| <+135>: lea -0x40(%rbp),%rax  <+139>: mov %rax,%rsi  <+142>: lea 0x0(%rip),%rdi # 0x1b5 <\_Z25CheckUserPermissionAccessv+149>  <+149>: callq 0x1ba <\_Z25CheckUserPermissionAccessv+154> | Reads the user’s input to 0x0(%rip) or String password |
| <+154>: lea -0x40(%rbp),%rax  <+158>: lea 0x0(%rip),%rsi # 0x1c5 <\_Z25CheckUserPermissionAccessv+165> | Return to main |

### DisplayInfo Function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x0(%rip),%rsi # 0x24c <\_Z11DisplayInfov+11>  <+11>: lea 0x0(%rip),%rdi # 0x253 <\_Z11DisplayInfov+18>  <+18>: callq 0x258 <\_Z11DisplayInfov+23> | Outputs the message “Client’s name Service Selected (1 = Brokerage, 2 = Retirement) |
| <+23>: mov %rax,%rdx  <+26>: mov 0x0(%rip),%rax # 0x262 <\_Z11DisplayInfov+33>  <+33>: mov %rax,%rsi  <+36>: mov %rdx,%rdi  <+39>: callq 0x26d <\_Z11DisplayInfov+44>  <+44>: lea 0x0(%rip),%rsi # 0x274 <\_Z11DisplayInfov+51>  <+51>: lea 0x0(%rip),%rdi # 0x27b <\_Z11DisplayInfov+58>  <+58>: callq 0x280 <\_Z11DisplayInfov+63>  <+63>: lea 0x0(%rip),%rsi # 0x287 <\_Z11DisplayInfov+70>  <+70>: mov %rax,%rdi  <+73>: callq 0x28f <\_Z11DisplayInfov+78>  <+78>: lea 0x0(%rip),%rsi # 0x296 <\_Z11DisplayInfov+85>  <+85>: mov %rax,%rdi  <+88>: callq 0x29e <\_Z11DisplayInfov+93>  <+93>: mov %rax,%rdx  <+96>: mov 0x0(%rip),%eax # 0x2a7 <\_Z11DisplayInfov+102>  <+102>: mov %eax,%esi  <+104>: mov %rdx,%rdi  <+107>: callq 0x2b1 <\_Z11DisplayInfov+112> | Prints out the message “1. “ << name1 << “ selected option ” << num1; |
| <+112>: mov %rax,%rdx  <+115>: mov 0x0(%rip),%rax # 0x2bb <\_Z11DisplayInfov+122>  <+122>: mov %rax,%rsi  <+125>: mov %rdx,%rdi  <+128>: callq 0x2c6 <\_Z11DisplayInfov+133>  <+133>: lea 0x0(%rip),%rsi # 0x2cd <\_Z11DisplayInfov+140>  <+140>: lea 0x0(%rip),%rdi # 0x2d4 <\_Z11DisplayInfov+147>  <+147>: callq 0x2d9 <\_Z11DisplayInfov+152>  <+152>: lea 0x0(%rip),%rsi # 0x2e0 <\_Z11DisplayInfov+159>  <+159>: mov %rax,%rdi  <+162>: callq 0x2e8 <\_Z11DisplayInfov+167>  <+167>: lea 0x0(%rip),%rsi # 0x2ef <\_Z11DisplayInfov+174>  <+174>: mov %rax,%rdi  <+177>: callq 0x2f7 <\_Z11DisplayInfov+182>  <+182>: mov %rax,%rdx  <+185>: mov 0x0(%rip),%eax # 0x300 <\_Z11DisplayInfov+191>  <+191>: mov %eax,%esi  <+193>: mov %rdx,%rdi  <+196>: callq 0x30a <\_Z11DisplayInfov+201> | Prints out the message “2. “ << name2 << “ selected option ” << num2; |
| <+201>: mov %rax,%rdx  <+204>: mov 0x0(%rip),%rax # 0x314 <\_Z11DisplayInfov+211>  <+211>: mov %rax,%rsi  <+214>: mov %rdx,%rdi  <+217>: callq 0x31f <\_Z11DisplayInfov+222>  <+222>: lea 0x0(%rip),%rsi # 0x326 <\_Z11DisplayInfov+229>  <+229>: lea 0x0(%rip),%rdi # 0x32d <\_Z11DisplayInfov+236>  <+236>: callq 0x332 <\_Z11DisplayInfov+241>  <+241>: lea 0x0(%rip),%rsi # 0x339 <\_Z11DisplayInfov+248>  <+248>: mov %rax,%rdi  <+251>: callq 0x341 <\_Z11DisplayInfov+256>  <+256>: lea 0x0(%rip),%rsi # 0x348 <\_Z11DisplayInfov+263>  <+263>: mov %rax,%rdi  <+266>: callq 0x350 <\_Z11DisplayInfov+271>  <+271>: mov %rax,%rdx  <+274>: mov 0x0(%rip),%eax # 0x359 <\_Z11DisplayInfov+280>  <+280>: mov %eax,%esi  <+282>: mov %rdx,%rdi  <+285>: callq 0x363 <\_Z11DisplayInfov+290> | Prints out the message “3. “ << name3 << “ selected option ” << num3; |
| <+290>: mov %rax,%rdx  <+293>: mov 0x0(%rip),%rax # 0x36d <\_Z11DisplayInfov+300>  <+300>: mov %rax,%rsi  <+303>: mov %rdx,%rdi  <+306>: callq 0x378 <\_Z11DisplayInfov+311>  <+311>: lea 0x0(%rip),%rsi # 0x37f <\_Z11DisplayInfov+318>  <+318>: lea 0x0(%rip),%rdi # 0x386 <\_Z11DisplayInfov+325>  <+325>: callq 0x38b <\_Z11DisplayInfov+330>  <+330>: lea 0x0(%rip),%rsi # 0x392 <\_Z11DisplayInfov+337>  <+337>: mov %rax,%rdi  <+340>: callq 0x39a <\_Z11DisplayInfov+345>  <+345>: lea 0x0(%rip),%rsi # 0x3a1 <\_Z11DisplayInfov+352>  <+352>: mov %rax,%rdi  <+355>: callq 0x3a9 <\_Z11DisplayInfov+360>  <+360>: mov %rax,%rdx  <+363>: mov 0x0(%rip),%eax # 0x3b2 <\_Z11DisplayInfov+369>  <+369>: mov %eax,%esi  <+371>: mov %rdx,%rdi  <+374>: callq 0x3bc <\_Z11DisplayInfov+379> | Prints out the message “4. “ << name4 << “ selected option ” << num4; |
| <+379>: mov %rax,%rdx  <+382>: mov 0x0(%rip),%rax # 0x3c6 <\_Z11DisplayInfov+389>  <+389>: mov %rax,%rsi  <+392>: mov %rdx,%rdi  <+395>: callq 0x3d1 <\_Z11DisplayInfov+400>  <+400>: lea 0x0(%rip),%rsi # 0x3d8 <\_Z11DisplayInfov+407>  <+407>: lea 0x0(%rip),%rdi # 0x3df <\_Z11DisplayInfov+414>  <+414>: callq 0x3e4 <\_Z11DisplayInfov+419>  <+419>: lea 0x0(%rip),%rsi # 0x3eb <\_Z11DisplayInfov+426>  <+426>: mov %rax,%rdi  <+429>: callq 0x3f3 <\_Z11DisplayInfov+434>  <+434>: lea 0x0(%rip),%rsi # 0x3fa <\_Z11DisplayInfov+441>  <+441>: mov %rax,%rdi  <+444>: callq 0x402 <\_Z11DisplayInfov+449>  <+449>: mov %rax,%rdx  <+452>: mov 0x0(%rip),%eax # 0x40b <\_Z11DisplayInfov+458>  <+458>: mov %eax,%esi  <+460>: mov %rdx,%rdi  <+463>: callq 0x415 <\_Z11DisplayInfov+468> | Prints out the message “5. “ << name5 << “ selected option ” << num5; |
| <+468>: mov %rax,%rdx  <+471>: mov 0x0(%rip),%rax # 0x41f <\_Z11DisplayInfov+478>  <+478>: mov %rax,%rsi  <+481>: mov %rdx,%rdi  <+484>: callq 0x42a <\_Z11DisplayInfov+489>  <+489>: nop  <+490>: pop %rbp  <+491>: retq | Returns to main of program |