# CS 410 Project One Proficiency Test

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

### “main” function” (Details Below)

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| Dump of assembler code for function main: |  |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x5eb(%rip),%rsi # 0x1440  <+11>: lea 0x201244(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+18>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt> | The main function is where the main greeting is printed to welcome the investment company. |
| <+23>: callq 0xf6a <\_Z25CheckUserPermissionAccessv>  <+28>: mov %eax,0x201494(%rip) # 0x202300 <answer>  <+34>: mov 0x20148e(%rip),%eax # 0x202300 <answer>  <+40>: cmp $0x1,%eax  <+43>: je 0xe8a <main+64>  <+45>: lea 0x5f2(%rip),%rsi # 0x1470  <+52>: lea 0x20121b(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+59>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+64>: mov 0x201470(%rip),%eax # 0x202300 <answer>  <+70>: cmp $0x1,%eax  <+73>: je 0xe97 <main+77>  <+75>: jmp 0xe61 <main+23> | The program executes a while loop that calls the function “CheckUserPermissionAccess,” and then it returns the value stored.  If the total is equal to true, the program will continue, but if it returns false, then it uses cout to print “Invalid Password” and prompted to try again.  The program will loop back to call the same function as above, “CheckUserPermissionAccess.” |
| <+77>: lea 0x5f6(%rip),%rsi # 0x1494  <+84>: lea 0x2011fb(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+91>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+96>: lea 0x5ff(%rip),%rsi # 0x14b0  <+103>: lea 0x2011e8(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+110>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+115>: lea 0x614(%rip),%rsi # 0x14d8  <+122>: lea 0x2011d5(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+129>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+134>: lea 0x625(%rip),%rsi # 0x14fc  <+141>: lea 0x2011c2(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+148>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+153>: lea 0x20140a(%rip),%rsi # 0x2022f4 <choice>  <+160>: lea 0x2012cf(%rip),%rdi # 0x2021c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+167>: callq 0xc60 <\_ZNSirsERi@plt>  <+172>: lea 0x61d(%rip),%rsi # 0x151a  <+179>: lea 0x20119c(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+186>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+191>: mov %rax,%rdx  <+194>: mov 0x2013e2(%rip),%eax # 0x2022f4 <choice>  <+200>: mov %eax,%esi  <+202>: mov %rdx,%rdi  <+205>: callq 0xd00 <\_ZNSolsEi@plt>  <+210>: mov %rax,%rdx  <+213>: mov 0x2010aa(%rip),%rax # 0x201fd0  <+220>: mov %rax,%rsi  <+223>: mov %rdx,%rdi  <+226>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> | The following group of code prints out a few strings requesting the user to input their selection.  “What would you like to do?  “Display the client list” and provides a user input option as “enter 1.”  “Change a client’s choice” and provides a user input option as “enter 2.”  “Exit the program” and provides a user input option as “enter 3” which will terminate the program.  The method “cin” is called and stored. Using cout, the user input displays “You chose” and prints the option they selected. |
|  |  |
| <+231>: mov 0x2013bd(%rip),%eax # 0x2022f4 <choice>  <+237>: cmp $0x1,%eax  <+240>: jne 0xf43 <main+249>  <+242>: callq 0x108b <\_Z11DisplayInfov>  <+247>: jmp 0xf53 <main+265>  <+249>: mov 0x2013ab(%rip),%eax # 0x2022f4 <choice>  <+255>: cmp $0x2,%eax  <+258>: jne 0xf53 <main+265>  <+260>: callq 0x1277 <\_Z20ChangeCustomerChoicev>  <+265>: mov 0x20139b(%rip),%eax # 0x2022f4 <choice>  <+271>: cmp $0x3,%eax  <+274>: je 0xf63 <main+281>  <+276>: jmpq 0xe97 <main+77>  <+281>: mov $0x0,%eax  <+286>: pop %rbp  <+287>: retq | Depending on the user input, the selected function is called and displayed.  Option 1: DisplayInfo  Option 2: ChangeCustomerChoice  Option 3: Exit  If the user inputs option 3, the program will jump to the end and the program is terminated. If the user inputs any other option, the program jumps to the beginning to line 77 for the user to input a new selection. |
| End of assembler dump. |  |

### ChangeCustomerChoice function (Details Below)

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| Dump of assembler code for function \_Z20ChangeCustomerChoicev: |  |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x346(%rip),%rsi # 0x5555555555c8  <+11>: lea 0x200e17(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+18>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+23>: lea 0x201063(%rip),%rsi # 0x5555557562f8 <changechoice>  <+30>: lea 0x200f24(%rip),%rdi # 0x5555557561c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+37>: callq 0x555555554c60 <\_ZNSirsERi@plt> | When the function is called by user input, the string is printed that displays, “Enter the number of the client you wish to change.”  The “cin” method is used to request the input and the user input is then stored into the variable “changechoice. |
| <+42>: lea 0x358(%rip),%rsi # 0x555555555600  <+49>: lea 0x200df1(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+56>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+61>: lea 0x201041(%rip),%rsi # 0x5555557562fc <newservice>  <+68>: lea 0x200efe(%rip),%rdi # 0x5555557561c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+75>: callq 0x555555554c60 <\_ZNSirsERi@plt> | In this section, the user is prompted with a new string requesting the user to input the new option for the client’s service.  “Please enter the client’s new service choice” and then prompts the user to select 1 or 2, which is displayed as “(1 = Brokerage, 2 = Retirement).  The “cin” method is used and the value is stored in the variable “newservice.” |
| <+80>: mov 0x20102b(%rip),%eax # 0x5555557562f8 <changechoice>  <+86>: cmp $0x1,%eax  <+89>: jne 0x5555555552e0 <\_Z20ChangeCustomerChoicev+105>  <+91>: mov 0x201024(%rip),%eax # 0x5555557562fc <newservice>  <+97>: mov %eax,0x200d82(%rip) # 0x555555756060 <num1>  <+103>: jmp 0x555555555342 <\_Z20ChangeCustomerChoicev+203>  <+105>: mov 0x201012(%rip),%eax # 0x5555557562f8 <changechoice>  <+111>: cmp $0x2,%eax  <+114>: jne 0x5555555552f9 <\_Z20ChangeCustomerChoicev+130>  <+116>: mov 0x20100b(%rip),%eax # 0x5555557562fc <newservice>  <+122>: mov %eax,0x200d6d(%rip) # 0x555555756064 <num2>  <+128>: jmp 0x555555555342 <\_Z20ChangeCustomerChoicev+203>  <+130>: mov 0x200ff9(%rip),%eax # 0x5555557562f8 <changechoice>  <+136>: cmp $0x3,%eax  <+139>: jne 0x555555555312 <\_Z20ChangeCustomerChoicev+155>  <+141>: mov 0x200ff2(%rip),%eax # 0x5555557562fc <newservice>  <+147>: mov %eax,0x200d58(%rip) # 0x555555756068 <num3>  <+153>: jmp 0x555555555342 <\_Z20ChangeCustomerChoicev+203>  <+155>: mov 0x200fe0(%rip),%eax # 0x5555557562f8 <changechoice>  <+161>: cmp $0x4,%eax  <+164>: jne 0x55555555532b <\_Z20ChangeCustomerChoicev+180>  <+166>: mov 0x200fd9(%rip),%eax # 0x5555557562fc <newservice>  <+172>: mov %eax,0x200d43(%rip) # 0x55555575606c <num4>  <+178>: jmp 0x555555555342 <\_Z20ChangeCustomerChoicev+203>  <+180>: mov 0x200fc7(%rip),%eax # 0x5555557562f8 <changechoice>  <+186>: cmp $0x5,%eax  <+189>: jne 0x555555555342 <\_Z20ChangeCustomerChoicev+203>  <+191>: mov 0x200fc0(%rip),%eax # 0x5555557562fc <newservice>  <+197>: mov %eax,0x200d2e(%rip) # 0x555555756070 <num5>  <+203>: nop  <+204>: pop %rbp  <+205>: retq | In this section, the program verifies the value of the variable, “changechoice.” Depending on the value, the program jumps to the appropriate line of code within the assembly code.  The stored value is then verified and if it equals to one of the previous numbers, the new value is updated to the value stored in “newservice.”  The function ChangeCustomerChoice returns. |
| End of assembler dump. |  |

### CheckUserPermissonAccess Function (Details Below)

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| Dump of assembler code for function \_Z25CheckUserPermissionAccessv: |  |
| <+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 0x555555554d20 <\_ZNSaIcEC1Ev@plt>  <+36>: lea -0x45(%rbp),%rdx  <+40>: lea -0x40(%rbp),%rax  <+44>: lea 0x588(%rip),%rsi # 0x555555555525  <+51>: mov %rax,%rdi  <+54>: callq 0x555555554ce0 <\_ZNSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEEC1EPKcRKS3\_@plt>  <+59>: lea -0x45(%rbp),%rax  <+63>: mov %rax,%rdi  <+66>: callq 0x555555554cb0 <\_ZNSaIcED1Ev@plt>  <+71>: movl $0x0,-0x44(%rbp)  <+78>: lea 0x567(%rip),%rsi # 0x555555555526  <+85>: lea 0x2010da(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+92>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+97>: lea 0x20130e(%rip),%rsi # 0x5555557562e0 <username>  <+104>: lea 0x2011e7(%rip),%rdi # 0x5555557561c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+111>: callq 0x555555554c40 <\_ZStrsIcSt11char\_traitsIcEERSt13basic\_istreamIT\_T0\_ES6\_PS3\_@plt> | In this section the function “CheckUserPermissionAccess” is called. The user is requested to input their username.  The string printed is “Enter your username:” The user input is then stored in the variable “username”. |
| <+116>: lea 0x558(%rip),%rsi # 0x55555555553d  <+123>: lea 0x2010b4(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+130>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+135>: lea -0x40(%rbp),%rax  <+139>: mov %rax,%rsi  <+142>: lea 0x2011c1(%rip),%rdi # 0x5555557561c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+149>: callq 0x555555554cd0 <\_ZStrsIcSt11char\_traitsIcESaIcEERSt13basic\_istreamIT\_T0\_ES7\_RNSt7\_\_cxx1112basic\_stringIS4\_S5\_T1\_EE@plt> | In this section, the user is requested to input their password.  The string printed is “Enter your password:” The user input is then stored in the variable “password”. |
| <+154>: lea -0x40(%rbp),%rax  <+158>: lea 0x545(%rip),%rsi # 0x555555555554  <+165>: mov %rax,%rdi  <+168>: callq 0x555555554c50 <\_ZNKSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEE7compareEPKc@plt>  <+173>: mov %eax,-0x44(%rbp)  <+176>: cmpl $0x0,-0x44(%rbp)  <+180>: jne 0x555555555027 <\_Z25CheckUserPermissionAccessv+189>  <+182>: mov $0x1,%ebx  <+187>: jmp 0x55555555502c <\_Z25CheckUserPermissionAccessv+194>  <+189>: mov $0x2,%ebx  <+194>: lea -0x40(%rbp),%rax  <+198>: mov %rax,%rdi  <+201>: callq 0x555555554c70 <\_ZNSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEED1Ev@plt>  <+206>: mov %ebx,%eax  <+208>: mov -0x18(%rbp),%rcx  <+212>: xor %fs:0x28,%rcx  <+221>: je 0x555555555084 <\_Z25CheckUserPermissionAccessv+282>  <+223>: jmp 0x55555555507f <\_Z25CheckUserPermissionAccessv+277>  <+225>: mov %rax,%rbx  <+228>: lea -0x45(%rbp),%rax  <+232>: mov %rax,%rdi  <+235>: callq 0x555555554cb0 <\_ZNSaIcED1Ev@plt>  <+240>: mov %rbx,%rax  <+243>: mov %rax,%rdi  <+246>: callq 0x555555554d10 <\_Unwind\_Resume@plt>  <+251>: mov %rax,%rbx  <+254>: lea -0x40(%rbp),%rax  <+258>: mov %rax,%rdi  <+261>: callq 0x555555554c70 <\_ZNSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEED1Ev@plt>  <+266>: mov %rbx,%rax  <+269>: mov %rax,%rdi  <+272>: callq 0x555555554d10 <\_Unwind\_Resume@plt>  <+277>: callq 0x555555554cc0 <\_\_stack\_chk\_fail@plt>  <+282>: add $0x48,%rsp  <+286>: pop %rbx  <+287>: pop %rbp  <+288>: retq | In this section, the values that have been stored above are compared to the values stored in the system to verify authentication.  If the values equal true, then the program continues. If the values equal false, then the program will return to the previous function for the user to input the username and password. |
| End of assembler dump. |  |

### DisplayInfo Function (Details Below)

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| Dump of assembler code for function \_Z11DisplayInfov: |  |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x4c2(%rip),%rsi # 0x555555555558  <+11>: lea 0x201003(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+18>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt> | In this section, the function “DisplayInfo” is called. The program prints the clients name and the service selected.  The string printed is “Client's Name Service Selected (1 = Brokerage, 2 = Retirement)” |
| <+23>: mov %rax,%rdx  <+26>: mov 0x200f24(%rip),%rax # 0x555555755fd0  <+33>: mov %rax,%rsi  <+36>: mov %rdx,%rdi  <+39>: callq 0x555555554ca0 <\_ZNSolsEPFRSoS\_E@plt>  <+44>: lea 0x4de(%rip),%rsi # 0x55555555559c  <+51>: lea 0x200fdb(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+58>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+63>: lea 0x200f3f(%rip),%rsi # 0x555555756010 <name1>  <+70>: mov %rax,%rdi  <+73>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+78>: lea 0x4c0(%rip),%rsi # 0x5555555555a0  <+85>: mov %rax,%rdi  <+88>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+93>: mov %rax,%rdx  <+96>: mov 0x200f6f(%rip),%eax # 0x555555756060 <num1>  <+102>: mov %eax,%esi  <+104>: mov %rdx,%rdi  <+107>: callq 0x555555554d00 <\_ZNSolsEi@plt>  <+112>: mov %rax,%rdx  <+115>: mov 0x200ecb(%rip),%rax # 0x555555755fd0  <+122>: mov %rax,%rsi  <+125>: mov %rdx,%rdi  <+128>: callq 0x555555554ca0 <\_ZNSolsEPFRSoS\_E@plt>  <+133>: lea 0x49b(%rip),%rsi # 0x5555555555b2  <+140>: lea 0x200f82(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+147>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+152>: lea 0x200ef6(%rip),%rsi # 0x555555756020 <name2>  <+159>: mov %rax,%rdi  <+162>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+167>: lea 0x467(%rip),%rsi # 0x5555555555a0  <+174>: mov %rax,%rdi  <+177>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+182>: mov %rax,%rdx  <+185>: mov 0x200f1a(%rip),%eax # 0x555555756064 <num2>  <+191>: mov %eax,%esi  <+193>: mov %rdx,%rdi  <+196>: callq 0x555555554d00 <\_ZNSolsEi@plt>  <+201>: mov %rax,%rdx  <+204>: mov 0x200e72(%rip),%rax # 0x555555755fd0  <+211>: mov %rax,%rsi  <+214>: mov %rdx,%rdi  <+217>: callq 0x555555554ca0 <\_ZNSolsEPFRSoS\_E@plt>  <+222>: lea 0x446(%rip),%rsi # 0x5555555555b6  <+229>: lea 0x200f29(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+236>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+241>: lea 0x200ead(%rip),%rsi # 0x555555756030 <name3>  <+248>: mov %rax,%rdi  <+251>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+256>: lea 0x40e(%rip),%rsi # 0x5555555555a0  <+263>: mov %rax,%rdi  <+266>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+271>: mov %rax,%rdx  <+274>: mov 0x200ec5(%rip),%eax # 0x555555756068 <num3>  <+280>: mov %eax,%esi  <+282>: mov %rdx,%rdi  <+285>: callq 0x555555554d00 <\_ZNSolsEi@plt>  <+290>: mov %rax,%rdx  <+293>: mov 0x200e19(%rip),%rax # 0x555555755fd0  <+300>: mov %rax,%rsi  <+303>: mov %rdx,%rdi  <+306>: callq 0x555555554ca0 <\_ZNSolsEPFRSoS\_E@plt>  <+311>: lea 0x3f1(%rip),%rsi # 0x5555555555ba  <+318>: lea 0x200ed0(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+325>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+330>: lea 0x200e64(%rip),%rsi # 0x555555756040 <name4>  <+337>: mov %rax,%rdi  <+340>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+345>: lea 0x3b5(%rip),%rsi # 0x5555555555a0  <+352>: mov %rax,%rdi  <+355>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+360>: mov %rax,%rdx  <+363>: mov 0x200e70(%rip),%eax # 0x55555575606c <num4>  <+369>: mov %eax,%esi  <+371>: mov %rdx,%rdi  <+374>: callq 0x555555554d00 <\_ZNSolsEi@plt>  <+379>: mov %rax,%rdx  <+382>: mov 0x200dc0(%rip),%rax # 0x555555755fd0  <+389>: mov %rax,%rsi  <+392>: mov %rdx,%rdi  <+395>: callq 0x555555554ca0 <\_ZNSolsEPFRSoS\_E@plt>  <+400>: lea 0x39c(%rip),%rsi # 0x5555555555be  <+407>: lea 0x200e77(%rip),%rdi # 0x5555557560a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+414>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+419>: lea 0x200e1b(%rip),%rsi # 0x555555756050 <name5>  <+426>: mov %rax,%rdi  <+429>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+434>: lea 0x35c(%rip),%rsi # 0x5555555555a0  <+441>: mov %rax,%rdi  <+444>: callq 0x555555554c90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+449>: mov %rax,%rdx  <+452>: mov 0x200e1b(%rip),%eax # 0x555555756070 <num5>  <+458>: mov %eax,%esi  <+460>: mov %rdx,%rdi  <+463>: callq 0x555555554d00 <\_ZNSolsEi@plt>  <+468>: mov %rax,%rdx  <+471>: mov 0x200d67(%rip),%rax # 0x555555755fd0  <+478>: mov %rax,%rsi  <+481>: mov %rdx,%rdi  <+484>: callq 0x555555554ca0 <\_ZNSolsEPFRSoS\_E@plt>  <+489>: nop  <+490>: pop %rbp  <+491>: retq | In this section, the names of each client are printed. The 5 clients printed shows the selected service value that is currently stored.  The client names are called and printed using the “std::cout” method that prints the client name as well as the selected option that is stored with the each variable previously declared.  For example, the following is printed:  Client's Name Service Selected (1 = Brokerage, 2 = Retirement)  1. Bob Jones selected option 1  2. Sarah Davis selected option 2  3. Amy Friendly selected option 1  4. Johnny Smith selected option 1  5. Carol Spears selected option 2  Once the information is displayed to the user, the function returns. |
| End of assembler dump. |  |