# 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 0x5eb(%rip),%rsi # 0x1440 "Hello! Welcome to our Investment Company\n"  <+11>: lea 0x201244(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+18>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt> | Sets up the stack and outputs the string "Hello! Welcome to our Investment Company\n" |
| <+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 "Invalid Password. Please try again\n"  <+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 function CheckUserPermissionAccess() is called and the return value is stored in variable answer. If answer is equal to 1 the program continues. If it is not equal to one it prints "Invalid Password. Please try again\n" and loops, calling the CheckUserPermissionAccess function. |
| <+77>: lea 0x5f6(%rip),%rsi # 0x1494 "What would you like to do?\n"  <+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 "DISPLAY the client list (enter 1)\n"  <+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 "CHANGE a client's choice (enter 2)\n"  <+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 "Exit the program.. (enter 3)\n"  <+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> <std::istream::operator>>(int&)@plt>  <+172>: lea 0x61d(%rip),%rsi # 0x151a "You chose "  <+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> <std::ostream::operator<<(int)@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> <std::ostream::operator<<(std::ostream& (\*)(std::ostream&))@plt> | The following strings are outputted to the terminal: "What would you like to do?\n", "DISPLAY the client list (enter 1)\n", "CHANGE a client's choice (enter 2)\n", "Exit the program.. (enter 3)\n". Then cin is called and the input is stored in the int choice variable. The following is then printed to the terminal: "You chose " << choice << "\n". |
|  |  |
| <+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 | If the value for choice equals 1 then the function DisplayInfo is called. If choice equals 2 then the function ChangeCustomerChoice is called. If choice equals 3 then the program jumps to line 265 and the program ends. If choice does not equal 3 then the program jumps back to line 77 in the previous block of assembly code. |

### ChangeCustomerChoice function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x346(%rip),%rsi # 0x15c8 "Enter the number of the client that you wish to change\n"  <+11>: lea 0x200e17(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+18>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+23>: lea 0x201063(%rip),%rsi # 0x2022f8 <changechoice> ""  <+30>: lea 0x200f24(%rip),%rdi # 0x2021c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+37>: callq 0xc60 <\_ZNSirsERi@plt> <std::istream::operator>>(int&)@plt> | The string "Enter the number of the client that you wish to change\n" is outputted to the terminal and cin then takes the user input and stores it in int variable changechoice. |
| <+42>: lea 0x358(%rip),%rsi # 0x1600 "Please enter the client's new service choice (1 = Brokerage, 2 = Retirement)\n"  <+49>: lea 0x200df1(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+56>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+61>: lea 0x201041(%rip),%rsi # 0x2022fc <newservice> ""  <+68>: lea 0x200efe(%rip),%rdi # 0x2021c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+75>: callq 0xc60 <\_ZNSirsERi@plt> <std::istream::operator>>(int&)@plt> | The string " Please enter the client's new service choice (1 = Brokerage, 2 = Retirement)\n" is outputted to the terminal and cin then takes the user input and stores it in int variable newservice. |
| <+80>: mov 0x20102b(%rip),%eax # 0x2022f8 <changechoice> ""  <+86>: cmp $0x1,%eax  <+89>: jne 0x12e0 <\_Z20ChangeCustomerChoicev+105>  <+91>: mov 0x201024(%rip),%eax # 0x2022fc <newservice> ""  <+97>: mov %eax,0x200d82(%rip) # 0x202060 <num1> "\001"  <+103>: jmp 0x1342 <\_Z20ChangeCustomerChoicev+203>  <+105>: mov 0x201012(%rip),%eax # 0x2022f8 <changechoice> ""  <+111>: cmp $0x2,%eax  <+114>: jne 0x12f9 <\_Z20ChangeCustomerChoicev+130>  <+116>: mov 0x20100b(%rip),%eax # 0x2022fc <newservice> ""  <+122>: mov %eax,0x200d6d(%rip) # 0x202064 <num2> "\002"  <+128>: jmp 0x1342 <\_Z20ChangeCustomerChoicev+203>  <+130>: mov 0x200ff9(%rip),%eax # 0x2022f8 <changechoice> ""  <+136>: cmp $0x3,%eax  <+139>: jne 0x1312 <\_Z20ChangeCustomerChoicev+155>  <+141>: mov 0x200ff2(%rip),%eax # 0x2022fc <newservice> ""  <+147>: mov %eax,0x200d58(%rip) # 0x202068 <num3> "\001"  <+153>: jmp 0x1342 <\_Z20ChangeCustomerChoicev+203>  <+155>: mov 0x200fe0(%rip),%eax # 0x2022f8 <changechoice> ""  <+161>: cmp $0x4,%eax  <+164>: jne 0x132b <\_Z20ChangeCustomerChoicev+180>  <+166>: mov 0x200fd9(%rip),%eax # 0x2022fc <newservice> ""  <+172>: mov %eax,0x200d43(%rip) # 0x20206c <num4> "\001"  <+178>: jmp 0x1342 <\_Z20ChangeCustomerChoicev+203>  <+180>: mov 0x200fc7(%rip),%eax # 0x2022f8 <changechoice> ""  <+186>: cmp $0x5,%eax  <+189>: jne 0x1342 <\_Z20ChangeCustomerChoicev+203>  <+191>: mov 0x200fc0(%rip),%eax # 0x2022fc <newservice> ""  <+197>: mov %eax,0x200d2e(%rip) # 0x202070 <num5> "\002"  <+203>: nop  <+204>: pop %rbp  <+205>: retq | The program checks if changechoice is equal to 1, 2, 3, 4, or 5 and jumps to the respective line in the assembly code. If changechoice equals one of the previous numbers, then it updates the value stored at the respective num variable (ie num1 for 1) to what was inputted for newservice. The function then returns. |

### 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 0xd20 <\_ZNSaIcEC1Ev@plt> #<std::allocator<char>::allocator()@plt>  <+36>: lea -0x45(%rbp),%rdx  <+40>: lea -0x40(%rbp),%rax  <+44>: lea 0x588(%rip),%rsi # 0x1525 "" Empty string  <+51>: mov %rax,%rdi  <+54>: callq 0xce0 <\_ZNSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEEC1EPKcRKS3\_@plt>  <+59>: lea -0x45(%rbp),%rax  <+63>: mov %rax,%rdi  <+66>: callq 0xcb0 <\_ZNSaIcED1Ev@plt> <std::allocator<char>::~allocator()@plt>  <+71>: movl $0x0,-0x44(%rbp)  <+78>: lea 0x567(%rip),%rsi # 0x1526 "Enter your username: \n"  <+85>: lea 0x2010da(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+92>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+97>: lea 0x20130e(%rip),%rsi # 0x2022e0 <username>  <+104>: lea 0x2011e7(%rip),%rdi # 0x2021c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+111>: callq 0xc40 <\_ZStrsIcSt11char\_traitsIcEERSt13basic\_istreamIT\_T0\_ES6\_PS3\_@plt> | The stack is initilized and the string "Enter your username: \n" is printed to the terminal. Input is then taken from the user and stored into string variable username. |
| <+116>: lea 0x558(%rip),%rsi # 0x153d "Enter your password: \n"  <+123>: lea 0x2010b4(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+130>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+135>: lea -0x40(%rbp),%rax  <+139>: mov %rax,%rsi  <+142>: lea 0x2011c1(%rip),%rdi # 0x2021c0 <\_ZSt3cin@@GLIBCXX\_3.4>  <+149>: callq 0xcd0 cd0 <std::basic\_istream<char, std::char\_traits<char> >& std::operator>><char, std::char\_traits<char>, std::allocator<char> >(std::basic\_istream<char, std::char\_traits<char> >&, std::\_\_cxx11::basic\_string<char, std::char\_traits<char>, std::allocator<char> >&)@plt> <\_ZStrsIcSt11char\_traitsIcESaIcEERSt13basic\_istreamIT\_T0\_ES7\_RNSt7\_\_cxx1112basic\_stringIS4\_S5\_T1\_EE@plt> | The string "Enter your password: \n" is printed to the terminal and input is taken from the user and stored into string variable password. |
| <+154>: lea -0x40(%rbp),%rax  <+158>: lea 0x545(%rip),%rsi # 0x1554 123  <+165>: mov %rax,%rdi  <+168>: callq 0xc50 <std::\_\_cxx11::basic\_string<char, std::char\_traits<char>, std::allocator<char> >::compare(char const\*) const@plt> <\_ZNKSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEE7compareEPKc@plt>  <+173>: mov %eax,-0x44(%rbp)  <+176>: cmpl $0x0,-0x44(%rbp)  <+180>: jne 0x1027 <\_Z25CheckUserPermissionAccessv+189> # Jump if the password entered is not equal to 123  <+182>: mov $0x1,%ebx  <+187>: jmp 0x102c <\_Z25CheckUserPermissionAccessv+194> # Jumps by default if the password is equal  <+189>: mov $0x2,%ebx  <+194>: lea -0x40(%rbp),%rax  <+198>: mov %rax,%rdi  <+201>: callq 0xc70 <\_ZNSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEED1Ev@plt> std::\_\_cxx11::basic\_string<char, std::char\_traits<char>, std::allocator<char> >::~basic\_string()@plt>  <+206>: mov %ebx,%eax  <+208>: mov -0x18(%rbp),%rcx  <+212>: xor %fs:0x28,%rcx  <+221>: je 0x1084 <\_Z25CheckUserPermissionAccessv+282>  <+223>: jmp 0x107f <\_Z25CheckUserPermissionAccessv+277>  <+225>: mov %rax,%rbx  <+228>: lea -0x45(%rbp),%rax  <+232>: mov %rax,%rdi  <+235>: callq 0xcb0 <\_ZNSaIcED1Ev@plt> <std::allocator<char>::~allocator()@plt>  <+240>: mov %rbx,%rax  <+243>: mov %rax,%rdi  <+246>: callq 0xd10 <\_Unwind\_Resume@plt>  <+251>: mov %rax,%rbx  <+254>: lea -0x40(%rbp),%rax  <+258>: mov %rax,%rdi  <+261>: callq 0xc70 <\_ZNSt7\_\_cxx1112basic\_stringIcSt11char\_traitsIcESaIcEED1Ev@plt> <std::\_\_cxx11::basic\_string<char, std::char\_traits<char>, std::allocator<char> >::~basic\_string()@plt>  <+266>: mov %rbx,%rax  <+269>: mov %rax,%rdi  <+272>: callq 0xd10 <\_Unwind\_Resume@plt>  <+277>: callq 0xcc0 <\_\_stack\_chk\_fail@plt>  <+282>: add $0x48,%rsp  <+286>: pop %rbx  <+287>: pop %rbp  <+288>: retq | The string compare function is used to compare the values stored in variable password and string “123”. If the value for password is equal to “123” then the function returns 1. If it does not equal “123” then it returns 2. |

### DisplayInfo Function

| **Assembly Code Block** | **Explanation of Functionality** |
| --- | --- |
| <+0>: push %rbp  <+1>: mov %rsp,%rbp  <+4>: lea 0x4c2(%rip),%rsi # 0x1558 "Client's Name Service Selected (1 = Brokerage, 2 = Retirement)"  <+11>: lea 0x201003(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+18>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt> | The string “Client's Name Service Selected (1 = Brokerage, 2 = Retirement)\n” is printed to the terminal. |
| <+23>: mov %rax,%rdx  <+26>: mov 0x200f24(%rip),%rax # 0x201fd0 ""  <+33>: mov %rax,%rsi  <+36>: mov %rdx,%rdi  <+39>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> #std::ostream::operator<<(std::ostream& (\*)(std::ostream&))  <+44>: lea 0x4de(%rip),%rsi # 0x159c "1. "  <+51>: lea 0x200fdb(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+58>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+63>: lea 0x200f3f(%rip),%rsi # 0x202010 <name1>: "Bob Jones"  <+70>: mov %rax,%rdi  <+73>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+78>: lea 0x4c0(%rip),%rsi # 0x15a0 " selected option "  <+85>: mov %rax,%rdi  <+88>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+93>: mov %rax,%rdx  <+96>: mov 0x200f6f(%rip),%eax # 0x202060 <num1>: "\001"  <+102>: mov %eax,%esi  <+104>: mov %rdx,%rdi  <+107>: callq 0xd00 <\_ZNSolsEi@plt> #std::ostream::operator<<(int)  <+112>: mov %rax,%rdx  <+115>: mov 0x200ecb(%rip),%rax # 0x201fd0 ""  <+122>: mov %rax,%rsi  <+125>: mov %rdx,%rdi  <+128>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> #std::ostream::operator<<(std::ostream& (\*)(std::ostream&))  <+133>: lea 0x49b(%rip),%rsi # 0x15b2 "2. "  <+140>: lea 0x200f82(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+147>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+152>: lea 0x200ef6(%rip),%rsi # 0x202020 <name2>: "Sarah Davis"  <+159>: mov %rax,%rdi  <+162>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+167>: lea 0x467(%rip),%rsi # 0x15a0 " selected option "  <+174>: mov %rax,%rdi  <+177>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+182>: mov %rax,%rdx  <+185>: mov 0x200f1a(%rip),%eax # 0x202064 <num2>: "\002"  <+191>: mov %eax,%esi  <+193>: mov %rdx,%rdi  <+196>: callq 0xd00 <\_ZNSolsEi@plt> #std::ostream::operator<<(int)  <+201>: mov %rax,%rdx  <+204>: mov 0x200e72(%rip),%rax # 0x201fd0 ""  <+211>: mov %rax,%rsi  <+214>: mov %rdx,%rdi  <+217>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> #std::ostream::operator<<(std::ostream& (\*)(std::ostream&))  <+222>: lea 0x446(%rip),%rsi # 0x15b6 "3. "  <+229>: lea 0x200f29(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+236>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+241>: lea 0x200ead(%rip),%rsi # 0x202030 <name3>: "Amy Friendly"  <+248>: mov %rax,%rdi  <+251>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+256>: lea 0x40e(%rip),%rsi # 0x15a0 " selected option "  <+263>: mov %rax,%rdi  <+266>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+271>: mov %rax,%rdx  <+274>: mov 0x200ec5(%rip),%eax # 0x202068 <num3>: "\001"  <+280>: mov %eax,%esi  <+282>: mov %rdx,%rdi  <+285>: callq 0xd00 <\_ZNSolsEi@plt> #std::ostream::operator<<(int)  <+290>: mov %rax,%rdx  <+293>: mov 0x200e19(%rip),%rax # 0x201fd0 ""  <+300>: mov %rax,%rsi  <+303>: mov %rdx,%rdi  <+306>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> #std::ostream::operator<<(std::ostream& (\*)(std::ostream&))  <+311>: lea 0x3f1(%rip),%rsi # 0x15ba "4. "  <+318>: lea 0x200ed0(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+325>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+330>: lea 0x200e64(%rip),%rsi # 0x202040 <name4>: "Johnny Smith"  <+337>: mov %rax,%rdi  <+340>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+345>: lea 0x3b5(%rip),%rsi # 0x15a0 " selected option "  <+352>: mov %rax,%rdi  <+355>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+360>: mov %rax,%rdx  <+363>: mov 0x200e70(%rip),%eax # 0x20206c <num4>: "\001"  <+369>: mov %eax,%esi  <+371>: mov %rdx,%rdi  <+374>: callq 0xd00 <\_ZNSolsEi@plt> #std::ostream::operator<<(int)  <+379>: mov %rax,%rdx  <+382>: mov 0x200dc0(%rip),%rax # 0x201fd0 ""  <+389>: mov %rax,%rsi  <+392>: mov %rdx,%rdi  <+395>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> #std::ostream::operator<<(std::ostream& (\*)(std::ostream&))  <+400>: lea 0x39c(%rip),%rsi # 0x15be "5. "  <+407>: lea 0x200e77(%rip),%rdi # 0x2020a0 <\_ZSt4cout@@GLIBCXX\_3.4>  <+414>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+419>: lea 0x200e1b(%rip),%rsi # 0x202050 <name5>: "Carol Spears"  <+426>: mov %rax,%rdi  <+429>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+434>: lea 0x35c(%rip),%rsi # 0x15a0 " selected option "  <+441>: mov %rax,%rdi  <+444>: callq 0xc90 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  <+449>: mov %rax,%rdx  <+452>: mov 0x200e1b(%rip),%eax # 0x202070 <num5>: "\002"  <+458>: mov %eax,%esi  <+460>: mov %rdx,%rdi  <+463>: callq 0xd00 <\_ZNSolsEi@plt> #std::ostream::operator<<(int)  <+468>: mov %rax,%rdx  <+471>: mov 0x200d67(%rip),%rax # 0x201fd0 ""  <+478>: mov %rax,%rsi  <+481>: mov %rdx,%rdi  <+484>: callq 0xca0 <\_ZNSolsEPFRSoS\_E@plt> #std::ostream::operator<<(std::ostream& (\*)(std::ostream&))  <+489>: nop  <+490>: pop %rbp  <+491>: retq | The following code block prints out the 5 users in the system and the selected service they use that is stored in their respective num variable (ie user number 1 has num1 as their selected service variable). Each user has their own line and follows a template which is printed to the terminal. Here is the cout expression for user one as an example, std::cout << "1. " << "Bob Jones" << " selected option " << num1 << "\n";. The function then returns once all of the users have been printed to the terminal. |