# **CS 410 Binary to C++ With Security Vulnerabilities Activity Template**

# **Paloma Rodriguez CS410 2/13/24**

**Step 1:** Convert the binary file to assembly code.

**Step 2:** Explain the functionality of the blocks of assembly code.

| **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- |
| 0000000000000000 <\_Z11DisplayMenuv>:  0: 55 push %rbp  1: 48 89 e5 mov %rsp,%rbp  4: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # b <\_Z11DisplayMenuv+0xb>  b: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 12 <\_Z11DisplayMenuv+0x12>  12: e8 00 00 00 00 callq 17 <\_Z11DisplayMenuv+0x17>  17: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 1e <\_Z11DisplayMenuv+0x1e>  1e: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 25 <\_Z11DisplayMenuv+0x25>  25: e8 00 00 00 00 callq 2a <\_Z11DisplayMenuv+0x2a>  2a: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 31 <\_Z11DisplayMenuv+0x31>  31: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 38 <\_Z11DisplayMenuv+0x38>  38: e8 00 00 00 00 callq 3d <\_Z11DisplayMenuv+0x3d>  3d: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 44 <\_Z11DisplayMenuv+0x44>  44: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 4b <\_Z11DisplayMenuv+0x4b>  4b: e8 00 00 00 00 callq 50 <\_Z11DisplayMenuv+0x50>  50: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 57 <\_Z11DisplayMenuv+0x57>  57: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 5e <\_Z11DisplayMenuv+0x5e>  5e: e8 00 00 00 00 callq 63 <\_Z11DisplayMenuv+0x63>  63: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 6a <\_Z11DisplayMenuv+0x6a>  6a: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 71 <\_Z11DisplayMenuv+0x71>  71: e8 00 00 00 00 callq 76 <\_Z11DisplayMenuv+0x76>  76: 90 nop  77: 5d pop %rbp  78: c3 retq | Display menu function, shows menu with multiple options, pushes base pointer into stack and sets up fram rsp, rbp. Calls functions/options to display menu options and then returns. |
| 0000000000000079 <main>:  79: 55 push %rbp  7a: 48 89 e5 mov %rsp,%rbp  7d: 48 83 ec 20 sub $0x20,%rsp  81: 64 48 8b 04 25 28 00 mov %fs:0x28,%rax  88: 00 00  8a: 48 89 45 f8 mov %rax,-0x8(%rbp)  8e: 31 c0 xor %eax,%eax  90: c7 45 ec 00 00 00 00 movl $0x0,-0x14(%rbp)  97: 8b 45 ec mov -0x14(%rbp),%eax  9a: 83 f8 05 cmp $0x5,%eax  9d: 0f 84 65 02 00 00 je 308 <main+0x28f>  a3: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # aa <main+0x31>  aa: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # b1 <main+0x38>  b1: e8 00 00 00 00 callq b6 <main+0x3d>  b6: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # bd <main+0x44>  bd: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # c4 <main+0x4b>  c4: e8 00 00 00 00 callq c9 <main+0x50>  c9: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # d0 <main+0x57>  d0: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # d7 <main+0x5e>  d7: e8 00 00 00 00 callq dc <main+0x63>  dc: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # e3 <main+0x6a>  e3: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # ea <main+0x71>  ea: e8 00 00 00 00 callq ef <main+0x76>  ef: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # f6 <main+0x7d>  f6: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # fd <main+0x84>  fd: e8 00 00 00 00 callq 102 <main+0x89>  102: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 109 <main+0x90>  109: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 110 <main+0x97>  110: e8 00 00 00 00 callq 115 <main+0x9c>  115: 48 8d 45 ec lea -0x14(%rbp),%rax  119: 48 89 c6 mov %rax,%rsi  11c: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 123 <main+0xaa>  123: e8 00 00 00 00 callq 128 <main+0xaf>  128: 8b 45 ec mov -0x14(%rbp),%eax  12b: 83 f8 01 cmp $0x1,%eax  12e: 0f 85 95 00 00 00 jne 1c9 <main+0x150>  134: 48 8d 45 f0 lea -0x10(%rbp),%rax  138: 48 89 c6 mov %rax,%rsi  13b: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 142 <main+0xc9>  142: e8 00 00 00 00 callq 147 <main+0xce>  147: 48 89 c2 mov %rax,%rdx  14a: 48 8d 45 f4 lea -0xc(%rbp),%rax  14e: 48 89 c6 mov %rax,%rsi  151: 48 89 d7 mov %rdx,%rdi  154: e8 00 00 00 00 callq 159 <main+0xe0>  159: 8b 45 f0 mov -0x10(%rbp),%eax  15c: 89 c6 mov %eax,%esi  15e: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 165 <main+0xec>  165: e8 00 00 00 00 callq 16a <main+0xf1>  16a: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 171 <main+0xf8>  171: 48 89 c7 mov %rax,%rdi  174: e8 00 00 00 00 callq 179 <main+0x100>  179: 48 89 c2 mov %rax,%rdx  17c: 8b 45 f4 mov -0xc(%rbp),%eax  17f: 89 c6 mov %eax,%esi  181: 48 89 d7 mov %rdx,%rdi  184: e8 00 00 00 00 callq 189 <main+0x110>  189: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 190 <main+0x117>  190: 48 89 c7 mov %rax,%rdi  193: e8 00 00 00 00 callq 198 <main+0x11f>  198: 48 89 c1 mov %rax,%rcx  19b: 8b 55 f0 mov -0x10(%rbp),%edx  19e: 8b 45 f4 mov -0xc(%rbp),%eax  1a1: 29 c2 sub %eax,%edx  1a3: 89 d0 mov %edx,%eax  1a5: 89 c6 mov %eax,%esi  1a7: 48 89 cf mov %rcx,%rdi  1aa: e8 00 00 00 00 callq 1af <main+0x136>  1af: 48 89 c2 mov %rax,%rdx  1b2: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 1b9 <main+0x140>  1b9: 48 89 c6 mov %rax,%rsi  1bc: 48 89 d7 mov %rdx,%rdi  1bf: e8 00 00 00 00 callq 1c4 <main+0x14b>  1c4: e9 ce fe ff ff jmpq 97 <main+0x1e>  1c9: 8b 45 ec mov -0x14(%rbp),%eax  1cc: 83 f8 02 cmp $0x2,%eax  1cf: 0f 85 93 00 00 00 jne 268 <main+0x1ef>  1d5: 48 8d 45 f0 lea -0x10(%rbp),%rax  1d9: 48 89 c6 mov %rax,%rsi  1dc: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 1e3 <main+0x16a>  1e3: e8 00 00 00 00 callq 1e8 <main+0x16f>  1e8: 48 89 c2 mov %rax,%rdx  1eb: 48 8d 45 f4 lea -0xc(%rbp),%rax  1ef: 48 89 c6 mov %rax,%rsi  1f2: 48 89 d7 mov %rdx,%rdi  1f5: e8 00 00 00 00 callq 1fa <main+0x181>  1fa: 8b 45 f0 mov -0x10(%rbp),%eax  1fd: 89 c6 mov %eax,%esi  1ff: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 206 <main+0x18d>  206: e8 00 00 00 00 callq 20b <main+0x192>  20b: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 212 <main+0x199>  212: 48 89 c7 mov %rax,%rdi  215: e8 00 00 00 00 callq 21a <main+0x1a1>  21a: 48 89 c2 mov %rax,%rdx  21d: 8b 45 f4 mov -0xc(%rbp),%eax  220: 89 c6 mov %eax,%esi  222: 48 89 d7 mov %rdx,%rdi  225: e8 00 00 00 00 callq 22a <main+0x1b1>  22a: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 231 <main+0x1b8>  231: 48 89 c7 mov %rax,%rdi  234: e8 00 00 00 00 callq 239 <main+0x1c0>  239: 48 89 c1 mov %rax,%rcx  23c: 8b 55 f0 mov -0x10(%rbp),%edx  23f: 8b 45 f4 mov -0xc(%rbp),%eax  242: 01 d0 add %edx,%eax  244: 89 c6 mov %eax,%esi  246: 48 89 cf mov %rcx,%rdi  249: e8 00 00 00 00 callq 24e <main+0x1d5>  24e: 48 89 c2 mov %rax,%rdx  251: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 258 <main+0x1df>  258: 48 89 c6 mov %rax,%rsi  25b: 48 89 d7 mov %rdx,%rdi  25e: e8 00 00 00 00 callq 263 <main+0x1ea>  263: e9 2f fe ff ff jmpq 97 <main+0x1e>  268: 8b 45 ec mov -0x14(%rbp),%eax  26b: 83 f8 03 cmp $0x3,%eax  26e: 0f 85 23 fe ff ff jne 97 <main+0x1e>  274: 48 8d 45 f0 lea -0x10(%rbp),%rax  278: 48 89 c6 mov %rax,%rsi  27b: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 282 <main+0x209>  282: e8 00 00 00 00 callq 287 <main+0x20e>  287: 48 89 c2 mov %rax,%rdx  28a: 48 8d 45 f4 lea -0xc(%rbp),%rax  28e: 48 89 c6 mov %rax,%rsi  291: 48 89 d7 mov %rdx,%rdi  294: e8 00 00 00 00 callq 299 <main+0x220>  299: 8b 45 f0 mov -0x10(%rbp),%eax  29c: 89 c6 mov %eax,%esi  29e: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 2a5 <main+0x22c>  2a5: e8 00 00 00 00 callq 2aa <main+0x231>  2aa: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 2b1 <main+0x238>  2b1: 48 89 c7 mov %rax,%rdi  2b4: e8 00 00 00 00 callq 2b9 <main+0x240>  2b9: 48 89 c2 mov %rax,%rdx  2bc: 8b 45 f4 mov -0xc(%rbp),%eax  2bf: 89 c6 mov %eax,%esi  2c1: 48 89 d7 mov %rdx,%rdi  2c4: e8 00 00 00 00 callq 2c9 <main+0x250>  2c9: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 2d0 <main+0x257>  2d0: 48 89 c7 mov %rax,%rdi  2d3: e8 00 00 00 00 callq 2d8 <main+0x25f>  2d8: 48 89 c1 mov %rax,%rcx  2db: 8b 45 f0 mov -0x10(%rbp),%eax  2de: 8b 75 f4 mov -0xc(%rbp),%esi  2e1: 99 cltd  2e2: f7 fe idiv %esi  2e4: 89 c6 mov %eax,%esi  2e6: 48 89 cf mov %rcx,%rdi  2e9: e8 00 00 00 00 callq 2ee <main+0x275>  2ee: 48 89 c2 mov %rax,%rdx  2f1: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 2f8 <main+0x27f>  2f8: 48 89 c6 mov %rax,%rsi  2fb: 48 89 d7 mov %rdx,%rdi  2fe: e8 00 00 00 00 callq 303 <main+0x28a>  303: e9 8f fd ff ff jmpq 97 <main+0x1e>  308: b8 00 00 00 00 mov $0x0,%eax  30d: 48 8b 4d f8 mov -0x8(%rbp),%rcx  311: 64 48 33 0c 25 28 00 xor %fs:0x28,%rcx  318: 00 00  31a: 74 05 je 321 <main+0x2a8>  31c: e8 00 00 00 00 callq 321 <main+0x2a8>  321: c9 leaveq  322: c3 retq | Main block function, start of program and after input, saves variables into designated locations, calls other functions based on conditional, returns and ends. |
| 0000000000000323 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii>:  323: 55 push %rbp  324: 48 89 e5 mov %rsp,%rbp  327: 48 83 ec 10 sub $0x10,%rsp  32b: 89 7d fc mov %edi,-0x4(%rbp)  32e: 89 75 f8 mov %esi,-0x8(%rbp)  331: 83 7d fc 01 cmpl $0x1,-0x4(%rbp)  335: 75 32 jne 369 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x46>  337: 81 7d f8 ff ff 00 00 cmpl $0xffff,-0x8(%rbp)  33e: 75 29 jne 369 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x46>  340: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 347 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x24>  347: e8 00 00 00 00 callq 34c <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x29>  34c: 48 8d 15 00 00 00 00 lea 0x0(%rip),%rdx # 353 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x30>  353: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 35a <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x37>  35a: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 361 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x3e>  361: 48 89 c7 mov %rax,%rdi  364: e8 00 00 00 00 callq 369 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii+0x46>  369: 90 nop  36a: c9 leaveq  36b: c3 retq | Handles static initialization and destruction, pushes pointer rbp and sets up stack into rsp,rbp. Checks conditionals and saves variables and returns. |
| 000000000000036c <\_GLOBAL\_\_sub\_I\_\_Z11DisplayMenuv>:  36c: 55 push %rbp  36d: 48 89 e5 mov %rsp,%rbp  370: be ff ff 00 00 mov $0xffff,%esi  375: bf 01 00 00 00 mov $0x1,%edi  37a: e8 a4 ff ff ff callq 323 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii>  37f: 5d pop %rbp  380: c3 retq | Global initialization function, pushes pointer into rbp and sets up stack, cleans and then returns. |

**Step 3:** Convert the assembly code to binary.

**Step 4:** Convert the assembly code to C++ code.

| **Blocks of Assembly Code** | **C++ Code** |
| --- | --- |
| 0000000000000000 <\_Z11DisplayMenuv>:  0: 55 push %rbp  1: 48 89 e5 mov %rsp,%rbp  4: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # b <\_Z11DisplayMenuv+0xb>  b: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 12 <\_Z11DisplayMenuv+0x12>  12: e8 00 00 00 00 callq 17 <\_Z11DisplayMenuv+0x17>  17: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 1e <\_Z11DisplayMenuv+0x1e>  1e: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 25 <\_Z11DisplayMenuv+0x25>  25: e8 00 00 00 00 callq 2a <\_Z11DisplayMenuv+0x2a>  2a: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 31 <\_Z11DisplayMenuv+0x31>  31: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 38 <\_Z11DisplayMenuv+0x38>  38: e8 00 00 00 00 callq 3d <\_Z11DisplayMenuv+0x3d>  3d: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 44 <\_Z11DisplayMenuv+0x44>  44: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 4b <\_Z11DisplayMenuv+0x4b>  4b: e8 00 00 00 00 callq 50 <\_Z11DisplayMenuv+0x50>  50: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 57 <\_Z11DisplayMenuv+0x57>  57: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 5e <\_Z11DisplayMenuv+0x5e>  5e: e8 00 00 00 00 callq 63 <\_Z11DisplayMenuv+0x63>  63: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 6a <\_Z11DisplayMenuv+0x6a>  6a: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 71 <\_Z11DisplayMenuv+0x71>  71: e8 00 00 00 00 callq 76 <\_Z11DisplayMenuv+0x76>  76: 90 nop  77: 5d pop %rbp  78: c3 retq | void DisplayMenu() {  cout << " " << endl;  cout << "Subtract -" << endl;  cout << "Add -" << endl;  cout << "Divide -" << endl;  cout << "Exit -" << endl;  cout << " " << endl; |
| 0000000000000079 <main>:  79: 55 push %rbp  7a: 48 89 e5 mov %rsp,%rbp  7d: 48 83 ec 20 sub $0x20,%rsp  81: 64 48 8b 04 25 28 00 mov %fs:0x28,%rax  88: 00 00  8a: 48 89 45 f8 mov %rax,-0x8(%rbp)  8e: 31 c0 xor %eax,%eax  90: c7 45 ec 00 00 00 00 movl $0x0,-0x14(%rbp)  97: 8b 45 ec mov -0x14(%rbp),%eax  9a: 83 f8 05 cmp $0x5,%eax  9d: 0f 84 65 02 00 00 je 308 <main+0x28f>  a3: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # aa <main+0x31>  aa: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # b1 <main+0x38>  b1: e8 00 00 00 00 callq b6 <main+0x3d>  b6: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # bd <main+0x44>  bd: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # c4 <main+0x4b>  c4: e8 00 00 00 00 callq c9 <main+0x50>  c9: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # d0 <main+0x57>  d0: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # d7 <main+0x5e>  d7: e8 00 00 00 00 callq dc <main+0x63>  dc: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # e3 <main+0x6a>  e3: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # ea <main+0x71>  ea: e8 00 00 00 00 callq ef <main+0x76>  ef: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # f6 <main+0x7d>  f6: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # fd <main+0x84>  fd: e8 00 00 00 00 callq 102 <main+0x89>  102: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 109 <main+0x90>  109: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 110 <main+0x97>  110: e8 00 00 00 00 callq 115 <main+0x9c>  115: 48 8d 45 ec lea -0x14(%rbp),%rax  119: 48 89 c6 mov %rax,%rsi  11c: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 123 <main+0xaa>  123: e8 00 00 00 00 callq 128 <main+0xaf>  128: 8b 45 ec mov -0x14(%rbp),%eax  12b: 83 f8 01 cmp $0x1,%eax  12e: 0f 85 95 00 00 00 jne 1c9 <main+0x150>  134: 48 8d 45 f0 lea -0x10(%rbp),%rax  138: 48 89 c6 mov %rax,%rsi  13b: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 142 <main+0xc9>  142: e8 00 00 00 00 callq 147 <main+0xce>  147: 48 89 c2 mov %rax,%rdx  14a: 48 8d 45 f4 lea -0xc(%rbp),%rax  14e: 48 89 c6 mov %rax,%rsi  151: 48 89 d7 mov %rdx,%rdi  154: e8 00 00 00 00 callq 159 <main+0xe0>  159: 8b 45 f0 mov -0x10(%rbp),%eax  15c: 89 c6 mov %eax,%esi  15e: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 165 <main+0xec>  165: e8 00 00 00 00 callq 16a <main+0xf1>  16a: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 171 <main+0xf8>  171: 48 89 c7 mov %rax,%rdi  174: e8 00 00 00 00 callq 179 <main+0x100>  179: 48 89 c2 mov %rax,%rdx  17c: 8b 45 f4 mov -0xc(%rbp),%eax  17f: 89 c6 mov %eax,%esi  181: 48 89 d7 mov %rdx,%rdi  184: e8 00 00 00 00 callq 189 <main+0x110>  189: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 190 <main+0x117>  190: 48 89 c7 mov %rax,%rdi  193: e8 00 00 00 00 callq 198 <main+0x11f>  198: 48 89 c1 mov %rax,%rcx  19b: 8b 55 f0 mov -0x10(%rbp),%edx  19e: 8b 45 f4 mov -0xc(%rbp),%eax  1a1: 29 c2 sub %eax,%edx  1a3: 89 d0 mov %edx,%eax  1a5: 89 c6 mov %eax,%esi  1a7: 48 89 cf mov %rcx,%rdi  1aa: e8 00 00 00 00 callq 1af <main+0x136>  1af: 48 89 c2 mov %rax,%rdx  1b2: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 1b9 <main+0x140>  1b9: 48 89 c6 mov %rax,%rsi  1bc: 48 89 d7 mov %rdx,%rdi  1bf: e8 00 00 00 00 callq 1c4 <main+0x14b>  1c4: e9 ce fe ff ff jmpq 97 <main+0x1e>  1c9: 8b 45 ec mov -0x14(%rbp),%eax  1cc: 83 f8 02 cmp $0x2,%eax  1cf: 0f 85 93 00 00 00 jne 268 <main+0x1ef>  1d5: 48 8d 45 f0 lea -0x10(%rbp),%rax  1d9: 48 89 c6 mov %rax,%rsi  1dc: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 1e3 <main+0x16a>  1e3: e8 00 00 00 00 callq 1e8 <main+0x16f>  1e8: 48 89 c2 mov %rax,%rdx  1eb: 48 8d 45 f4 lea -0xc(%rbp),%rax  1ef: 48 89 c6 mov %rax,%rsi  1f2: 48 89 d7 mov %rdx,%rdi  1f5: e8 00 00 00 00 callq 1fa <main+0x181>  1fa: 8b 45 f0 mov -0x10(%rbp),%eax  1fd: 89 c6 mov %eax,%esi  1ff: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 206 <main+0x18d>  206: e8 00 00 00 00 callq 20b <main+0x192>  20b: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 212 <main+0x199>  212: 48 89 c7 mov %rax,%rdi  215: e8 00 00 00 00 callq 21a <main+0x1a1>  21a: 48 89 c2 mov %rax,%rdx  21d: 8b 45 f4 mov -0xc(%rbp),%eax  220: 89 c6 mov %eax,%esi  222: 48 89 d7 mov %rdx,%rdi  225: e8 00 00 00 00 callq 22a <main+0x1b1>  22a: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 231 <main+0x1b8>  231: 48 89 c7 mov %rax,%rdi  234: e8 00 00 00 00 callq 239 <main+0x1c0>  239: 48 89 c1 mov %rax,%rcx  23c: 8b 55 f0 mov -0x10(%rbp),%edx  23f: 8b 45 f4 mov -0xc(%rbp),%eax  242: 01 d0 add %edx,%eax  244: 89 c6 mov %eax,%esi  246: 48 89 cf mov %rcx,%rdi  249: e8 00 00 00 00 callq 24e <main+0x1d5>  24e: 48 89 c2 mov %rax,%rdx  251: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 258 <main+0x1df>  258: 48 89 c6 mov %rax,%rsi  25b: 48 89 d7 mov %rdx,%rdi  25e: e8 00 00 00 00 callq 263 <main+0x1ea>  263: e9 2f fe ff ff jmpq 97 <main+0x1e>  268: 8b 45 ec mov -0x14(%rbp),%eax  26b: 83 f8 03 cmp $0x3,%eax  26e: 0f 85 23 fe ff ff jne 97 <main+0x1e>  274: 48 8d 45 f0 lea -0x10(%rbp),%rax  278: 48 89 c6 mov %rax,%rsi  27b: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 282 <main+0x209>  282: e8 00 00 00 00 callq 287 <main+0x20e>  287: 48 89 c2 mov %rax,%rdx  28a: 48 8d 45 f4 lea -0xc(%rbp),%rax  28e: 48 89 c6 mov %rax,%rsi  291: 48 89 d7 mov %rdx,%rdi  294: e8 00 00 00 00 callq 299 <main+0x220>  299: 8b 45 f0 mov -0x10(%rbp),%eax  29c: 89 c6 mov %eax,%esi  29e: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 2a5 <main+0x22c>  2a5: e8 00 00 00 00 callq 2aa <main+0x231>  2aa: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 2b1 <main+0x238>  2b1: 48 89 c7 mov %rax,%rdi  2b4: e8 00 00 00 00 callq 2b9 <main+0x240>  2b9: 48 89 c2 mov %rax,%rdx  2bc: 8b 45 f4 mov -0xc(%rbp),%eax  2bf: 89 c6 mov %eax,%esi  2c1: 48 89 d7 mov %rdx,%rdi  2c4: e8 00 00 00 00 callq 2c9 <main+0x250>  2c9: 48 8d 35 00 00 00 00 lea 0x0(%rip),%rsi # 2d0 <main+0x257>  2d0: 48 89 c7 mov %rax,%rdi  2d3: e8 00 00 00 00 callq 2d8 <main+0x25f>  2d8: 48 89 c1 mov %rax,%rcx  2db: 8b 45 f0 mov -0x10(%rbp),%eax  2de: 8b 75 f4 mov -0xc(%rbp),%esi  2e1: 99 cltd  2e2: f7 fe idiv %esi  2e4: 89 c6 mov %eax,%esi  2e6: 48 89 cf mov %rcx,%rdi  2e9: e8 00 00 00 00 callq 2ee <main+0x275>  2ee: 48 89 c2 mov %rax,%rdx  2f1: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 2f8 <main+0x27f>  2f8: 48 89 c6 mov %rax,%rsi  2fb: 48 89 d7 mov %rdx,%rdi  2fe: e8 00 00 00 00 callq 303 <main+0x28a>  303: e9 8f fd ff ff jmpq 97 <main+0x1e>  308: b8 00 00 00 00 mov $0x0,%eax  30d: 48 8b 4d f8 mov -0x8(%rbp),%rcx  311: 64 48 33 0c 25 28 00 xor %fs:0x28,%rcx  318: 00 00  31a: 74 05 je 321 <main+0x2a8>  31c: e8 00 00 00 00 callq 321 <main+0x2a8>  321: c9 leaveq  322: c3 retq | int choice = 0;  Display();  cin >> choice;  while (choice != 4) {  int num1, num2;  cout << "Enter First Number: " << endl;  cin >> num1;  cout << "Enter Second Number: " << endl;  cin >> num2;  if (choice == 1) {  cout << num1 << " - " << num2 << " = " << num1 - num2 << endl;  }  else if (choice == 2) {  cout << num1 << " + " << num2 << " = " << num1 + num2 << endl;  }  else if (choice == 3) {  cout << num1 << " / " << num2 << " = " << (double)num1 / (double)num2 << endl;  }  else {  cout << "error" << endl;  }  } |
| 000000000000036c <\_GLOBAL\_\_sub\_I\_\_Z11DisplayMenuv>:  36c: 55 push %rbp  36d: 48 89 e5 mov %rsp,%rbp  370: be ff ff 00 00 mov $0xffff,%esi  375: bf 01 00 00 00 mov $0x1,%edi  37a: e8 a4 ff ff ff callq 323 <\_Z41\_\_static\_initialization\_and\_destruction\_0ii>  37f: 5d pop %rbp  380: c3 retq | Display();  cin >> choice;  }  } |