

Class: CS-362  
Term: Summer 2017  
Author: Jon-Eric Cook  
Date: July 30, 2017  
Assignment: #5

## **Re-factoring**

Per the instructions, a teammate's (Amelia Le) code was copied and put into a separate folder within my "projects/cookjone" folder. The name of the folder is "leameDominion". All my test code was copied pasted into leameDominion/dominion. Said test code is as follows: unittest1.c, unittest2.c, unittest3.c, unittest4.c, cardtest1.c, cardtest2.c, cardtest3.c, cardtest4.c, randomtestadventurer.c, randomtestcard1.c, and randomtestcard2.c. Once all the code was in place, the Makefile was modified to compile and run all the above files and have their outputs sent to assignment5tests.out.

Upon further inspection of Amelia's code, it was a pleasant surprise to see that she chose similar cards and functions as me to run her tests on. When it came to which cards she chose to introduce bugs into, again, there were a lot of similarities. In her play\_adventurer card function, she chose to introduce a bug that borderline couldn't be caught. The bug was as follows:

```
if (state->deckCount[currentPlayer] < 1)
*changed to*
if (state->deckCount[currentPlayer] < 2).
If the program has to cycle through the entire deck to find a treasure card, it misses the last
card.
```

This bug proved difficult to catch and sadly, it was not caught at all. I spoke with Amelia about this and she agreed that it was a hard bug to catch and that her randomtestadventurer.c only sometimes caught it.

Other than the above odd bug, all the other bugs were straightforward and easy to catch. Due to them being so straightforward, my test code was able to catch the bugs without too much refactoring. The one portion of my code that needed refactoring was for play\_council\_room. The bug in this function was that it skipped the first player, other than the current player, when it came time to drawing an extra card. This refactoring took place in cardtest3.c. The one change was simply making player1 the current player rather than player0. By this small refactoring, the bug in play\_council\_room was caught.

On the topic of maintainability, I believe my test suit is quite maintainable. When I set out to build my tests, I made sure that it would run every line of code in the FUT. By doing this, I was able to test each aspect of the function and also confirm the changes to the game state were correct. If the changes to the game state were wrong (a bug), it would be caught. Because I had this approach, I didn't have to refactor my test code a lot when it came to testing Amelia's code.

## **Bug-Reports**

The first bug that was found occurred in the play\_smithy function on line 696. This bug was found by setting up the game state and then calling cardEffect with smithy as the card argument. Once the cardEffect function returned, the hand count and deck count for each player was compared to the expected outcome. It was found that the bug caused the current

player to draw 4 cards instead of only 3. The root cause of this bug was that the for loop that was in charge of drawing the cards for the current player had its upper bound increased by one. The correct version and the buggy version of the code block can be seen below.

```
// draw 3 cards
for (i = 0; i < 3; i++)
{
    drawCard(currentPlayer, state);
}

// **BUG** draw 4 Cards
for (i = 0; i < 4; i++)
{
    drawCard(currentPlayer, state);
}
```

As a whole, I would not say this is an especially interesting bug. This type of bug could have simply been a slip of a finger, pressing 4 instead of 3.

The second bug that was found occurred in the play\_council\_room function on line 722. This bug was found by setting up the game state and then calling cardEffect with council\_room as the card argument. Once the cardEffect function returned, the hand count and deck count for each player was compared to the expected outcome. It was found that the bug caused the program to miss player 0 when it came time for each player, other than current player, to draw a card. The root cause of this bug was that the for loop that was in charge of drawing the cards for all others players, besides the current player, had its lower bound increased by one. The correct version and the buggy version of the code block can be seen below.

```
// Each player, besides the current player, draws a card
for (i = 0; i < state->numPlayers; i++)
{
    if (i != currentPlayer)
    {
        drawCard(i, state);
    }
}

// **BUG** Each player, besides the current player, draws a card
for (i = 1; i < state->numPlayers; i++)
{
    if (i != currentPlayer)
    {
        drawCard(i, state);
    }
}
```

As a whole, I would say this is a somewhat interesting bug. This type of bug might always not show up, let alone be caught. If for example, the current player in 0, then this bug would have no effect.

The third bug that was found occurred in the play\_village function on line 745. This bug was found by setting up the game state and then calling cardEffect with village as the card

argument. Once the cardEffect function returned, the number of actions for the game state was compared to the expected outcome. It was found that the bug caused the numActions count to one less than it was suppose to be. The root cause of this bug was that the variable numActions was only incremented once when it was suppose to incremented twice. The correct version and the buggy version of the code block can be seen below.

```
// +2 Actions
state->numActions = state->numActions + 2;

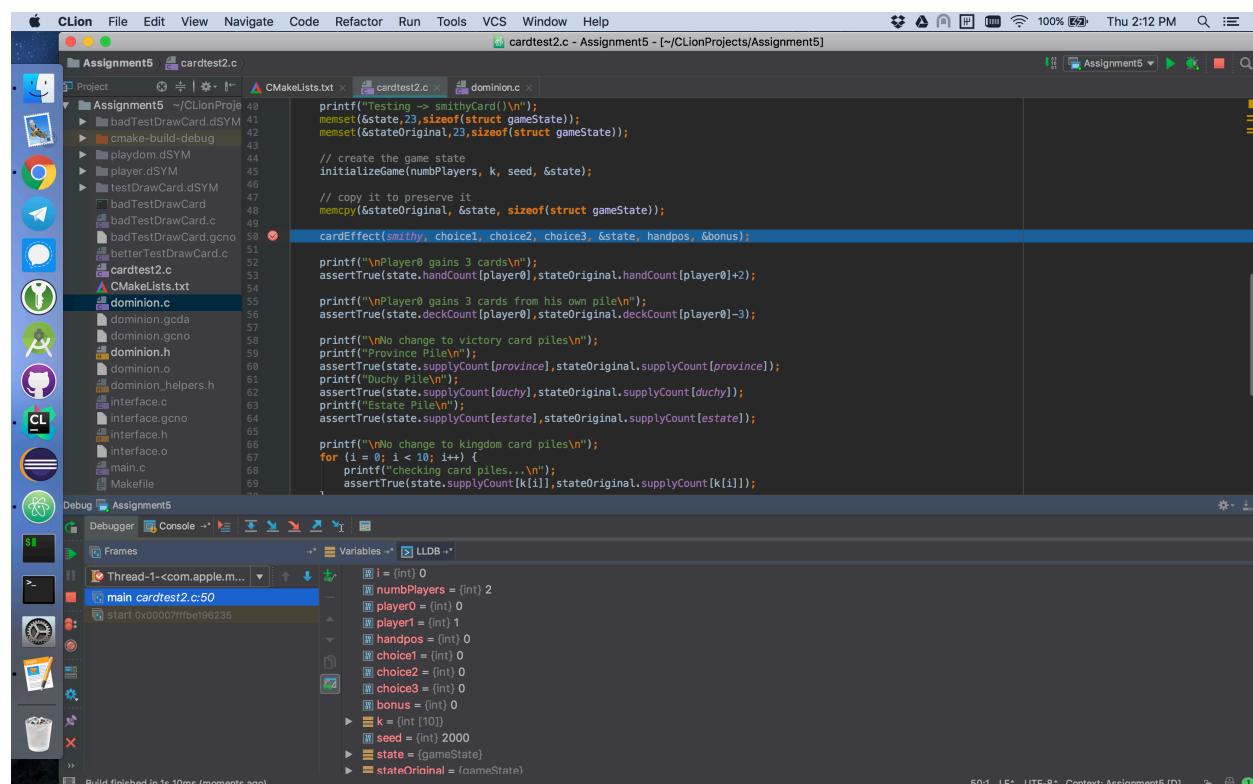
// **BUG** +1 Action
state->numActions = state->numActions + 1;
```

Again, as a whole, I would not say this is an especially interesting bug. This type of bug could have simply been a slip of a finger, pressing 1 instead of 2.

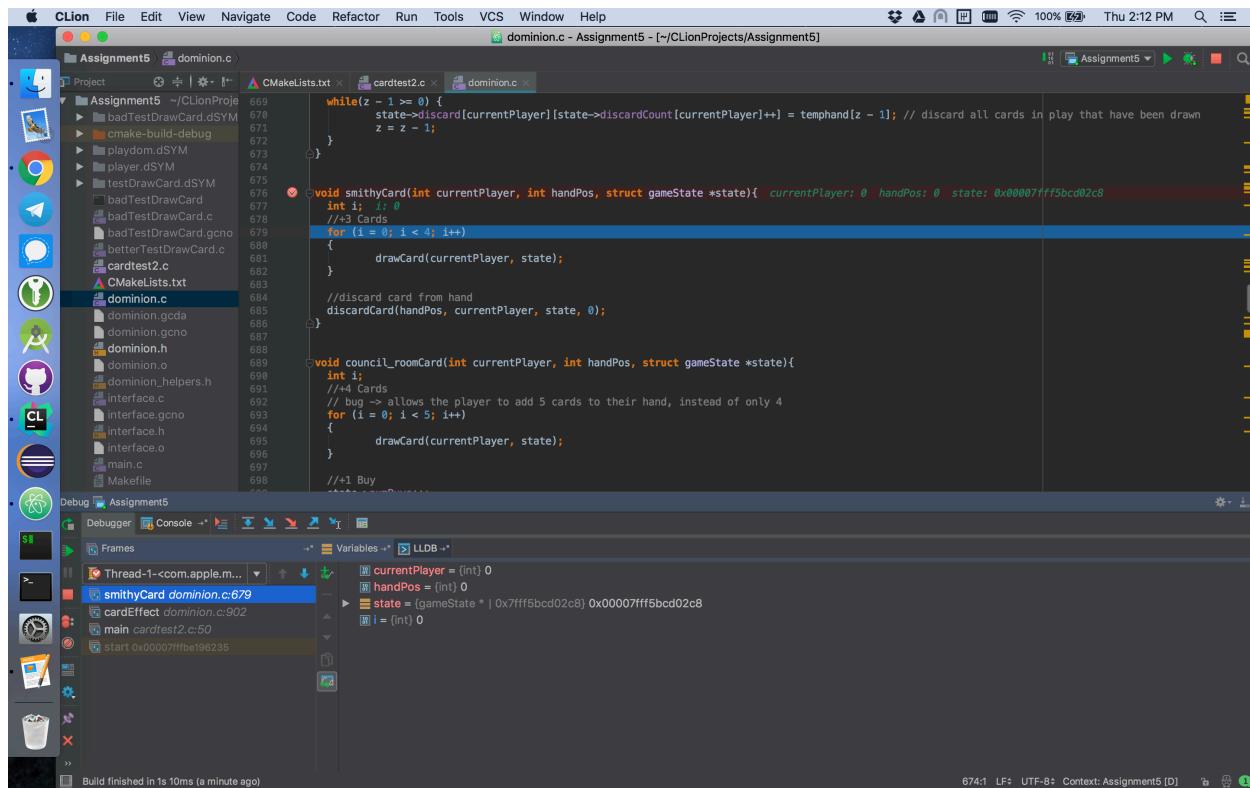
## Debugging

The bug report for my code, “BugsInTeammateCode.txt”, located in projects/cookjone/dominion, was the starting place for the process of identifying and fixing a bug in my own code. From line 1 to 19, Amelia Le identifies the failure of the smithyCard function. She outlines the expected outcome of the function -> “draw 3 cards”. She also outlines the actual outcome of the function -> “draw 4 cards”. Towards the bottom of the bug report, line 18, she indicates line 680 in my dominion.c file holds the error and also provides a snippet of the bug filled code. Based on this information, I knew where to look and how to debug it. Employing the debugging tools in CLion was the next step in fixing the bug.

Jumping into CLion, I put break points at line 50 in cardtest2.c, and line 676 in dominion.c. These two places were chosen because they were where cardEffect was called in the cardtest2.c and where the smithyCard functions started in dominion.c. By placing break points at these two places, it allowed me to step through the bug filled code and examine what was going on, line by line. Below is the starting place of the debugging process.



The next step was to step into the cardEffect function and make my way to the smithyCard function. Below the starting place of stepping through the smithyCard function.



CLion File Edit View Navigate Code Refactor Run Tools VCS Window Help

dominion.c - Assignment5 - [~/CLionProjects/Assignment5]

Assignment5 ~/CLionProj...

Project CMakeLists.txt cardtest2.c dominion.c

Assignments5 ~/CLionProj...

badTestDrawCard.dSYM cmake-build-debug playdom.dSYM player.dSYM testDrawCard.dSYM badTestDrawCard.c badTestDrawCard.gno betterTestDrawCard.c cardtest2.c CMakelists.txt dominion.c dominion.gcda dominion.gnco dominion.h dominion.o dominion\_helpers.h interface.c interface.gnco interface.h interface.o main.c Makefile

679 while(z - 1 >= 0) { state->discard[currentPlayer][state->discardCount[currentPlayer]]++ = tempHand[z - 1]; z = z - 1; }

680 } void smithyCard(int currentPlayer, int handPos, struct gameState \*state){ currentPlayer: 0 handPos: 0 state: 0x00007fff5bcd02c8

681 int i; i: 0 //+3 Cards

682 for (i = 0; i < 4; i++) { drawCard(currentPlayer, state); }

683 //discard card from hand

684 discardCard(handPos, currentPlayer, state, 0); }

685 void council\_roomCard(int currentPlayer, int handPos, struct gameState \*state){

686 int i; //+4 Cards

687 // bug -> allows the player to add 5 cards to their hand, instead of only 4

688 for (i = 0; i < 5; i++) { drawCard(currentPlayer, state); }

689 //+1 Buy

690 }

691 }

692 }

693 }

694 }

695 }

696 }

697 }

698 }

699 }

700 }

701 }

702 }

703 }

704 }

705 }

706 }

707 }

708 }

709 }

710 }

711 }

712 }

713 }

714 }

715 }

716 }

717 }

718 }

719 }

720 }

721 }

722 }

723 }

724 }

725 }

726 }

727 }

728 }

729 }

730 }

731 }

732 }

733 }

734 }

735 }

736 }

737 }

738 }

739 }

740 }

741 }

742 }

743 }

744 }

745 }

746 }

747 }

748 }

749 }

750 }

751 }

752 }

753 }

754 }

755 }

756 }

757 }

758 }

759 }

760 }

761 }

762 }

763 }

764 }

765 }

766 }

767 }

768 }

769 }

770 }

771 }

772 }

773 }

774 }

775 }

776 }

777 }

778 }

779 }

780 }

781 }

782 }

783 }

784 }

785 }

786 }

787 }

788 }

789 }

790 }

791 }

792 }

793 }

794 }

795 }

796 }

797 }

798 }

799 }

800 }

801 }

802 }

803 }

804 }

805 }

806 }

807 }

808 }

809 }

810 }

811 }

812 }

813 }

814 }

815 }

816 }

817 }

818 }

819 }

820 }

821 }

822 }

823 }

824 }

825 }

826 }

827 }

828 }

829 }

830 }

831 }

832 }

833 }

834 }

835 }

836 }

837 }

838 }

839 }

840 }

841 }

842 }

843 }

844 }

845 }

846 }

847 }

848 }

849 }

850 }

851 }

852 }

853 }

854 }

855 }

856 }

857 }

858 }

859 }

860 }

861 }

862 }

863 }

864 }

865 }

866 }

867 }

868 }

869 }

870 }

871 }

872 }

873 }

874 }

875 }

876 }

877 }

878 }

879 }

880 }

881 }

882 }

883 }

884 }

885 }

886 }

887 }

888 }

889 }

890 }

891 }

892 }

893 }

894 }

895 }

896 }

897 }

898 }

899 }

900 }

901 }

902 }

903 }

904 }

905 }

906 }

907 }

908 }

909 }

910 }

911 }

912 }

913 }

914 }

915 }

916 }

917 }

918 }

919 }

920 }

921 }

922 }

923 }

924 }

925 }

926 }

927 }

928 }

929 }

930 }

931 }

932 }

933 }

934 }

935 }

936 }

937 }

938 }

939 }

940 }

941 }

942 }

943 }

944 }

945 }

946 }

947 }

948 }

949 }

950 }

951 }

952 }

953 }

954 }

955 }

956 }

957 }

958 }

959 }

960 }

961 }

962 }

963 }

964 }

965 }

966 }

967 }

968 }

969 }

970 }

971 }

972 }

973 }

974 }

975 }

976 }

977 }

978 }

979 }

980 }

981 }

982 }

983 }

984 }

985 }

986 }

987 }

988 }

989 }

990 }

991 }

992 }

993 }

994 }

995 }

996 }

997 }

998 }

999 }

1000 }

1001 }

1002 }

1003 }

1004 }

1005 }

1006 }

1007 }

1008 }

1009 }

1010 }

1011 }

1012 }

1013 }

1014 }

1015 }

1016 }

1017 }

1018 }

1019 }

1020 }

1021 }

1022 }

1023 }

1024 }

1025 }

1026 }

1027 }

1028 }

1029 }

1030 }

1031 }

1032 }

1033 }

1034 }

1035 }

1036 }

1037 }

1038 }

1039 }

1040 }

1041 }

1042 }

1043 }

1044 }

1045 }

1046 }

1047 }

1048 }

1049 }

1050 }

1051 }

1052 }

1053 }

1054 }

1055 }

1056 }

1057 }

1058 }

1059 }

1060 }

1061 }

1062 }

1063 }

1064 }

1065 }

1066 }

1067 }

1068 }

1069 }

1070 }

1071 }

1072 }

1073 }

1074 }

1075 }

1076 }

1077 }

1078 }

1079 }

1080 }

1081 }

1082 }

1083 }

1084 }

1085 }

1086 }

1087 }

1088 }

1089 }

1090 }

1091 }

1092 }

1093 }

1094 }

1095 }

1096 }

1097 }

1098 }

1099 }

1100 }

1101 }

1102 }

1103 }

1104 }

1105 }

1106 }

1107 }

1108 }

1109 }

1110 }

1111 }

1112 }

1113 }

1114 }

1115 }

1116 }

1117 }

1118 }

1119 }

1120 }

1121 }

1122 }

1123 }

1124 }

1125 }

1126 }

1127 }

1128 }

1129 }

1130 }

1131 }

1132 }

1133 }

1134 }

1135 }

1136 }

1137 }

1138 }

1139 }

1140 }

1141 }

1142 }

1143 }

1144 }

1145 }

1146 }

1147 }

1148 }

1149 }

1150 }

1151 }

1152 }

1153 }

1154 }

1155 }

1156 }

1157 }

1158 }

1159 }

1160 }

1161 }

1162 }

1163 }

1164 }

1165 }

1166 }

1167 }

1168 }

1169 }

1170 }

1171 }

1172 }

1173 }

1174 }

1175 }

1176 }

1177 }

1178 }

1179 }

1180 }

1181 }

1182 }

1183 }

1184 }

1185 }

1186 }

1187 }

1188 }

1189 }

1190 }

1191 }

1192 }

1193 }

1194 }

1195 }

1196 }

1197 }

1198 }

1199 }

1200 }

1201 }

1202 }

1203 }

1204 }

1205 }

1206 }

1207 }

1208 }

1209 }

1210 }

1211 }

1212 }

1213 }

1214 }

1215 }

1216 }

1217 }

1218 }

1219 }

1220 }

1221 }

1222 }

1223 }

1224 }

1225 }

1226 }

1227 }

1228 }

1229 }

1230 }

1231 }

1232 }

1233 }

1234 }

1235 }

1236 }

1237 }

1238 }

1239 }

1240 }

1241 }

1242 }

1243 }

1244 }

1245 }

1246 }

1247 }

1248 }

1249 }

1250 }

1251 }

1252 }

1253 }

1254 }

1255 }

1256 }

1257 }

1258 }

1259 }

1260 }

1261 }

1262 }

1263 }

1264 }

1265 }

1266 }

1267 }

1268 }

1269 }

1270 }

1271 }

1272 }

1273 }

1274 }

1275 }

1276 }

1277 }

1278 }

1279 }

1280 }

1281 }

1282 }

1283 }

1284 }

1285 }

1286 }

1287 }

1288 }

1289 }

1290 }

1291 }

1292 }

1293 }

1294 }

1295 }

1296 }

1297 }

1298 }

1299 }

1300 }

1301 }

1302 }

1303 }

1304 }

1305 }

1306 }

1307 }

1308 }

1309 }

1310 }

1311 }

1312 }

1313 }

1314 }

1315 }

1316 }

1317 }

1318 }

1319 }

1320 }

1321 }

1322 }

1323 }

1324 }

1325 }

1326 }

1327 }

1328 }

1329 }

1330 }

1331 }

1332 }

1333 }

1334 }

1335 }

1336 }

1337 }

1338 }

1339 }

1340 }

1341 }

1342 }

1343 }

1344 }

1345 }

1346 }

1347 }

1348 }

1349 }

1350 }

1351 }

1352 }

1353 }

1354 }

1355 }

1356 }

1357 }

1358 }

1359 }

1360 }

1361 }

1362 }

1363 }

1364 }

1365 }

1366 }

1367 }

1368 }

1369 }

1370 }

1371 }

1372 }

1373 }

1374 }

1375 }

1376 }

1377 }

1378 }

1379 }

1380 }

1381 }

1382 }

1383 }

1384 }

1385 }

1386 }

1387 }

1388 }

1389 }

1390 }

1391 }

1392 }

1393 }

1394 }

1395 }

1396 }

1397 }

1398 }

1399 }

1400 }

1401 }

1402 }

1403 }

1404 }

1405 }

1406 }

1407 }

1408 }

1409 }

1410 }

1411 }

1412 }

1413 }

1414 }

1415 }

1416 }

1417 }

1418 }

1419 }

1420 }

1421 }

1422 }

1423 }

1424 }

1425 }

1426 }

1427 }

1428 }

1429 }

1430 }

1431 }

1432 }

1433 }

1434 }

1435 }

1436 }

1437 }

1438 }

1439 }

1440 }

1441 }

1442 }

1443 }

1444 }

1445 }

1446 }

1447 }

1448 }

1449 }

1450 }

1451 }

1452 }

1453 }

1454 }

1455 }

1456 }

1457 }

1458 }

1459 }

1460 }

1461 }

1462 }

1463 }

1464 }

1465 }

1466 }

1467 }

1468 }

1469 }

1470 }

1471 }

1472 }

1473 }

1474 }

1475 }

1476 }

1477 }

1478 }

1479 }

1480 }

1481 }

1482 }

1483 }

1484 }

1485 }

1486 }

1487 }

1488 }

1489 }

1490 }

1491 }

1492 }

1493 }

1494 }

1495 }

1496 }

1497 }

1498 }

1499 }

1500 }

1501 }

1502 }

1503 }

1504 }

1505 }

1506 }

1507 }

1508 }

1509 }

1510 }

1511 }

1512 }

1513 }

1514 }

1515 }

1516 }

1517 }

1518 }

1519 }

1520 }

1521 }

1522 }

1523 }

1524 }

1525 }

1526 }

1527 }

1528 }

1529 }

1530 }

1531 }

1532 }

1533 }

1534 }

1535 }

1536 }

1537 }

1538 }

1539 }

1540 }

1541 }

1542 }

1543 }

1544 }

1545 }

1546 }

1547 }

1548 }

1549 }

1550 }

1551 }

1552 }

1553 }

1554 }

1555 }

1556 }

1557 }

1558 }

1559 }

1560 }

1561 }

1562 }

1563 }

1564 }

1565 }

1566 }

1567 }

1568 }

1569 }

1570 }

1571 }

1572 }

1573 }

1574 }

1575 }

1576 }

1577 }

1578 }

1579 }

1580 }

1581 }

1582 }

1583 }

1584 }

1585 }

1586 }

1587 }

1588 }

1589 }

1590 }

1591 }

1592 }

1593 }

1594 }

1595 }

1596 }

1597 }

1598 }

1599 }

1600 }

1601 }

1602 }

1603 }

1604 }

1605 }

1606 }

1607 }

1608 }

1609 }

1610 }

1611 }

1612 }

1613 }

1614 }

1615 }

1616 }

1617 }

1618 }

1619 }

1620 }

1621 }

1622 }

1623 }

1624 }

1625 }

1626 }

1627 }

1628 }

1629 }

1630 }

1631 }

1632 }

1633 }

1634 }

1635 }

1636 }

1637 }

1638 }

1639 }

1640 }

1641 }

1642 }

1643 }

1644 }

1645 }

1646 }

1647 }

1648 }

1649 }

1650 }

1651 }

1652 }

1653 }

1654 }

1655 }

1656 }

1657 }

1658 }

1659 }

1660 }

1661 }

1662 }

1663 }

1664 }

1665 }

1666 }

1667 }

1668 }

1669 }

1670 }

1671 }

1672 }

1673 }

1674 }

1675 }

1676 }

1677 }

1678 }

1679 }

1680 }

1681 }

1682 }

1683 }

1684 }

1685 }

1686 }

1687 }

1688 }

1689 }

1690 }

1691 }

1692 }

1693 }

1694 }

1695 }

1696 }

1697 }

1698 }

1699 }

1700 }

1701 }

1702 }

1703 }

1704 }

1705 }

1706 }

1707 }

1708 }

1709 }

1710 }

1711 }

1712 }

1713 }

1714 }

1715 }

1716 }

1717 }

1718 }

1719 }

1720 }

1721 }

1722 }

1723 }

1724 }

1725 }

1726 }

1727 }

1728 }

1729 }

1730 }

1731 }

1732 }

1733 }

1734 }

1735 }

1736 }

1737 }

1738 }

1739 }

1740 }

1741 }

1742 }

1743 }

1744 }

1745 }

1746 }

1747 }

1748 }

1749 }

1750 }

1751 }

1752 }

1753 }

1754 }

1755 }

1756 }

1757 }

1758 }

1759 }

1760 }

1761 }

1762 }

1763 }

1764 }

1765 }

1766 }

1767 }

1768 }

1769 }

1770 }

1771 }

1772 }

1773 }

1774 }

1775 }

1776 }

1777 }

1778 }

1779 }

1780 }

1781 }

1782 }

1783 }

1784 }

1785 }

1786 }

1787 }

1788 }

1789 }

1790 }

1791 }

1792 }

1793 }

1794 }

1795 }

1796 }

1797 }

1798 }

1799 }

1800 }

1801 }

1802 }

1803 }

1804 }

1805 }

1806 }

1807 }

1808 }

1809 }

1810 }

1811 }

1812 }

1813 }

1814 }

1815 }

1816 }

1817 }

1818 }

1819 }

1820 }

1821 }

1822 }

1823 }

1824 }

1825 }

1826 }

1827 }

1828 }

1829 }

1830 }

1831 }

1832 }

1833 }

1834 }

1835 }

1836 }

1837 }

1838 }

1839 }

1840 }

1841 }

1842 }

1843 }

1844 }

1845 }

1846 }

1847 }

1848 }

1849 }

1850 }

1851 }

1852 }

1853 }

1854 }

1855 }

1856 }

1857 }

1858 }

1859 }

1860 }

1861 }

1862 }

1863 }

1864 }

1865 }

1866 }

1867 }

1868 }

1869 }

1870 }

1871 }

1872 }

1873 }

1874 }

1875 }

1876 }

1877 }

1878 }

1879 }

1880 }

1881 }

1882 }

1883 }

1884 }

1885 }

1886 }

1887 }

1888 }

1889 }

1890 }

1891 }

1892 }

1893 }

1894 }

1895 }

1896 }

1897 }

1898 }

1899 }

1900 }

1901 }

1902 }

1903 }

1904 }

1905 }

1906 }

1907 }

1908 }

1909 }

1910 }

1911 }

1912 }

1913 }

1914 }

1915 }

1916 }

1917 }

1918 }</

In the above photo, it can be seen that "i" incremented all the way up to 4, meaning that the for loop ran 4 times. Thus the drawCard function was executed 4 times. This is the bug that Amelie Le pointed out. Based on the above discovery, changes were made to fix the bug. The bug was fixed by simply changing the upper bound of the for loop on line 679 in dominion.c to 3 instead of 4. A new debugging session was started and below is the start of it.

```

Assignment5 - Assignment5 - [~/CLionProjects/Assignment5]
Project: Assignment5 - ~/CLionProjects/Assignment5
File: cardtest2.c
Line: 679

    if (i < 0) {
        state->discard[currentPlayer][state->discardCount[currentPlayer]] = tempHand[z - 1]; // discard all cards in play that have been drawn
        z = z - 1;
    }

    void smithyCard(int currentPlayer, int handPos, struct gameState *state){ currentPlayer: 0 handPos: 0 state: 0x00007fff5f1162c8
        int i;
        //+2 Cards
        for (i = 0; i < 3; i++)
        {
            drawCard(currentPlayer, state);
        }
        //discard card from hand
        discardCard(handPos, currentPlayer, state, 0);
    }

    void council_roomCard(int currentPlayer, int handPos, struct gameState *state){
        int i;
        //+4 Cards
        // bug -> allows the player to add 5 cards to their hand, instead of only 4
        for (i = 0; i < 5; i++)
        {
            drawCard(currentPlayer, state);
        }
        //+1 Buy
    }
}

```

Continuing on, below it can be seen that the upper bound was in fact changed.

```

Assignment5 - Assignment5 - [~/CLionProjects/Assignment5]
Project: Assignment5 - ~/CLionProjects/Assignment5
File: dominion.c
Line: 679

    if (i < 0) {
        state->discard[currentPlayer][state->discardCount[currentPlayer]] = tempHand[z - 1]; // discard all cards in play that have been drawn
        z = z - 1;
    }

    void smithyCard(int currentPlayer, int handPos, struct gameState *state){ currentPlayer: 0 handPos: 0 state: 0x00007fff5f1162c8
        int i;
        //+2 Cards
        for (i = 0; i < 3; i++)
        {
            drawCard(currentPlayer, state);
        }
        //discard card from hand
        discardCard(handPos, currentPlayer, state, 0);
    }

    void council_roomCard(int currentPlayer, int handPos, struct gameState *state){
        int i;
        //+4 Cards
        // bug -> allows the player to add 5 cards to their hand, instead of only 4
        for (i = 0; i < 5; i++)
        {
            drawCard(currentPlayer, state);
        }
        //+1 Buy
    }
}

```

In the photo below, it can be seen that "i" only incremented up to 3, meaning that the for loop ran 3 times. Thus the drawCard function was executed 3 times, the correct amount.

```
while(z - 1 >= 0) {
    state->discard[currentPlayer][state->discardCount[currentPlayer]]++ = tempHand[z - 1]; // discard all cards in play that have been drawn
    z = z - 1;
}

void smithyCard(int currentPlayer, int handPos, struct gameState *state){
    currentPlayer: 0 handPos: 0 state: 0x00007ffff5f1162c8
    int i; i: 3
    //+3 Cards
    for (i = 0; i < 3; i++) {
        drawCard(currentPlayer, state);
    }
    //discard card from hand
    discardCard(handPos, currentPlayer, state, 0);
}

void council_roomCard(int currentPlayer, int handPos, struct gameState *state){
    int i;
    //+4 Cards
    // bug -> allows the player to add 5 cards to their hand, instead of only 4
    for (i = 0; i < 5; i++) {
        drawCard(currentPlayer, state);
    }
    //+1 Buy
    //+1 Buy
```

Build finished in 999ms (a minute ago)

678:13 LF: UTF-8 Context: Assignment5 [D]