



Control Flow Graph Testing

Divyesh Radadiya (MT2021044)

Prashant Chaudhary (MT2021102)

Overview

The aim of this project is to perform Control Flow Graph testing on a sample code that covers the edge and prime path for all the functions in the code.

[Source code](#)

Basic Code Description

- We have made: Bank calculator, Health calculator, Scientific calculator and Conversion calculator. Each calculator has different functions related to their domains.
- Bank Calculator helps to find emi, simple interest, compound interest etc.
- Health calculator can calculate Body Mass Index, Body Fat Percentage etc.
- Scientific Calculator helps to perform scientific calculations such as finding factorial, square root, power etc.

Testing Strategy Used

Control Flow Graph

It is the graphical representation of control flow or computation during the execution of programs or applications. It is mostly used in static analysis and compiler applications. We have used control flow graph testing to test our code.

Testing Tool Used

- We have used JUnit to design our test cases. JUnit is a testing framework for Java programming language which is used for unique testing of code.
- <http://cs.gmu.edu/~offutt/softwaretest/forTRgeneration>

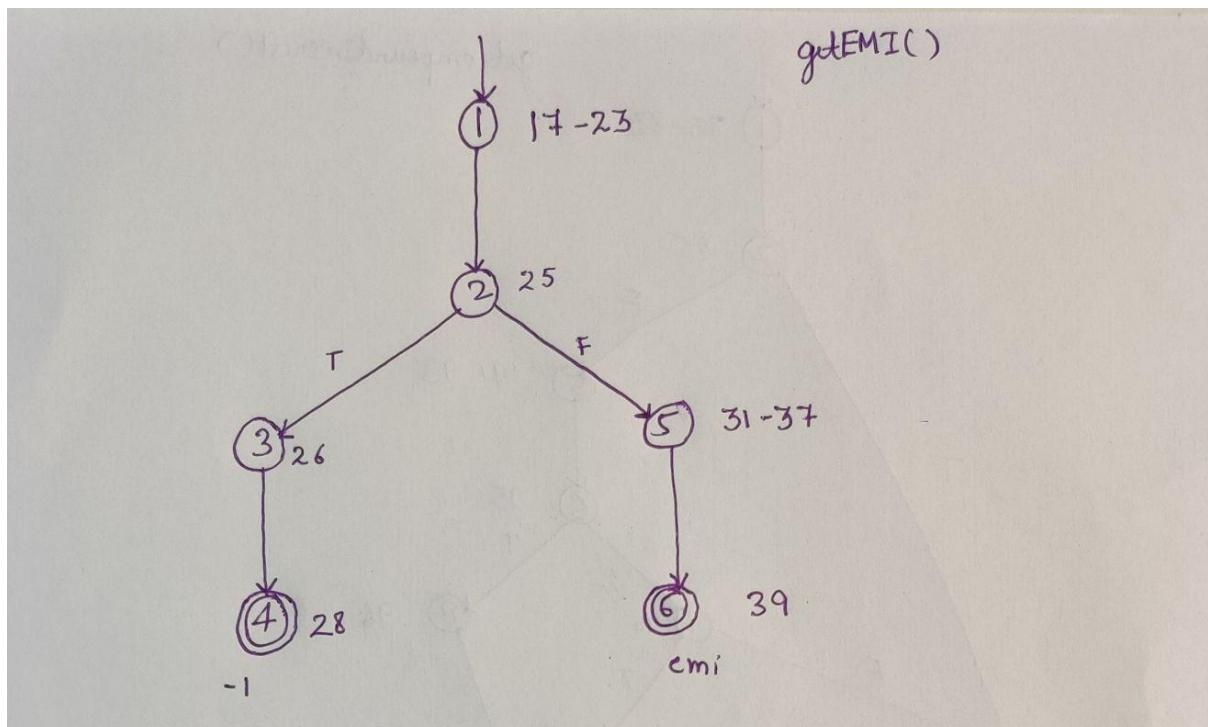
I. Bank Calculator

1. getEMI

- This function is used to get the emi

```
16     @RequestMapping(value = "/getEMI", method = RequestMethod.POST)
17     public double getEMI(@RequestBody Map<String, String> body){
18
19         int amount = Integer.parseInt(body.get("amount"));
20
21         int term = Integer.parseInt(body.get("term")); // month
22
23         double interestRate = Double.parseDouble( body.get("interestRate"));
24
25         if (amount < 0 || term < 0 || interestRate < 0) {...}
26
27         double convertedRateOfInterest = interestRate / (12 * 100);
28
29         double exponent = Math.pow((1 + convertedRateOfInterest), term);
30
31         double emi = amount * convertedRateOfInterest * exponent / (exponent -1);
32
33         logger.info("[EMI] - INPUT:" +amount+"^"+term+"^"+interestRate+ " , OUTPUT:" + emi);
34
35         return emi ;
36     }
```

CFG and Block Number



Testcase

Run: BankTest.EMI

Tests passed: 1 of 1 test - 1 sec 530 ms

BankTest (com.example.Softw.1 sec 530 ms) C:\Users\drada\jdks\openjdk-18.0.1.1\bin\java.exe ...
26/Nov/2022:15:27:01 +0530 [BankController.java] [INFO] BankController [EMI] - INPUT:1000^1^10.0 , OUTPUT:Invalid Input
26/Nov/2022:15:27:01 +0530 [BankController.java] [INFO] BankController [EMI] - INPUT:1000^5^10.0 , OUTPUT:205.027661850526

Process finished with exit code 0

Testpath

2 test paths are needed for Edge Coverage

[1,2,5,6]

[1,2,3,4]

2 test paths are needed for Prime Path Coverage

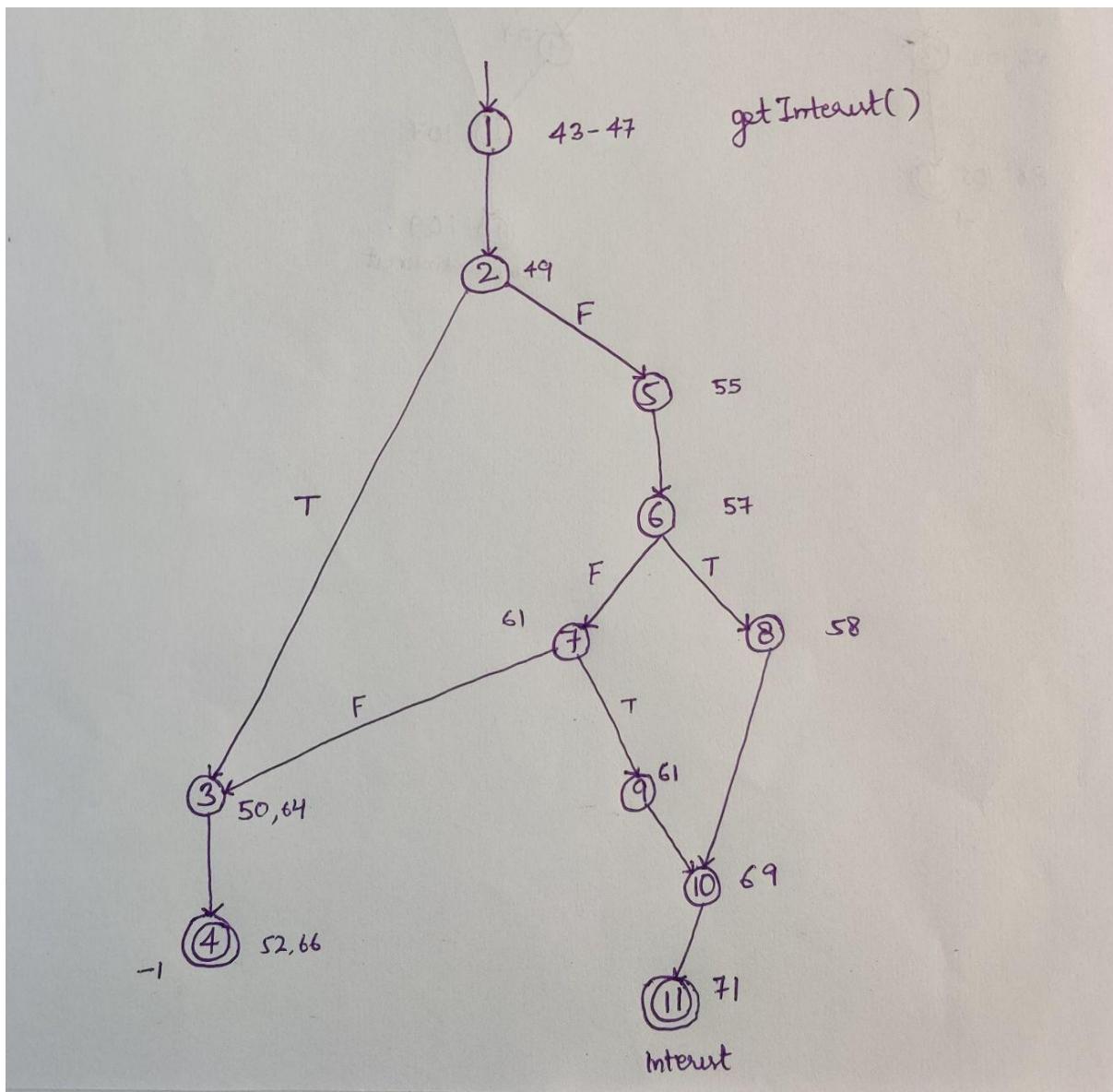
Test Paths	Test Requirements that are toured by test paths directly
[1,2,3,4]	[1,2,3,4]
[1,2,5,6]	[1,2,5,6]

2. getInterest

- This function is used to get the Interest

```
43  @
44
45     public double getInterest(@RequestBody Map<String, String> body){
46
47         int amount = Integer.parseInt(body.get("amount"));
48
49         if (amount < 0) {...}
50
51         double interest = 0;
52
53         if(acType=="NRI") {
54             interest=amount * 0.06;
55         }
56         else if(acType=="Normal") {
57             interest=amount * 0.04;
58         }
59         else {
60             logger.info("[Interest] - INPUT:" +amount+"^"+acType+ " , OUTPUT:" + "Invalid Input");
61
62             return -1;
63         }
64
65         logger.info("[Interest] - INPUT:" +amount+"^"+acType+ " , OUTPUT:" + interest);
66
67         return interest ;
68
69
70
71
72 }
```

CFG and Block Number



Testcase

Screenshot of a Java IDE showing the output of a test run. The test name is `BankTest.interest`. The output shows the command used to run the test and the log output:

```
Run: BankTest.interest X
  0 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 | 125 | 126 | 127 | 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 | 137 | 138 | 139 | 140 | 141 | 142 | 143 | 144 | 145 | 146 | 147 | 148 | 149 | 150 | 151 | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 | 160 | 161 | 162 | 163 | 164 | 165 | 166 | 167 | 168 | 169 | 170 | 171 | 172 | 173 | 174 | 175 | 176 | 177 | 178 | 179 | 180 | 181 | 182 | 183 | 184 | 185 | 186 | 187 | 188 | 189 | 190 | 191 | 192 | 193 | 194 | 195 | 196 | 197 | 198 | 199 | 200 | 201 | 202 | 203 | 204 | 205 | 206 | 207 | 208 | 209 | 210 | 211 | 212 | 213 | 214 | 215 | 216 | 217 | 218 | 219 | 220 | 221 | 222 | 223 | 224 | 225 | 226 | 227 | 228 | 229 | 230 | 231 | 232 | 233 | 234 | 235 | 236 | 237 | 238 | 239 | 240 | 241 | 242 | 243 | 244 | 245 | 246 | 247 | 248 | 249 | 250 | 251 | 252 | 253 | 254 | 255 | 256 | 257 | 258 | 259 | 260 | 261 | 262 | 263 | 264 | 265 | 266 | 267 | 268 | 269 | 270 | 271 | 272 | 273 | 274 | 275 | 276 | 277 | 278 | 279 | 280 | 281 | 282 | 283 | 284 | 285 | 286 | 287 | 288 | 289 | 290 | 291 | 292 | 293 | 294 | 295 | 296 | 297 | 298 | 299 | 300 | 301 | 302 | 303 | 304 | 305 | 306 | 307 | 308 | 309 | 310 | 311 | 312 | 313 | 314 | 315 | 316 | 317 | 318 | 319 | 320 | 321 | 322 | 323 | 324 | 325 | 326 | 327 | 328 | 329 | 330 | 331 | 332 | 333 | 334 | 335 | 336 | 337 | 338 | 339 | 340 | 341 | 342 | 343 | 344 | 345 | 346 | 347 | 348 | 349 | 350 | 351 | 352 | 353 | 354 | 355 | 356 | 357 | 358 | 359 | 360 | 361 | 362 | 363 | 364 | 365 | 366 | 367 | 368 | 369 | 370 | 371 | 372 | 373 | 374 | 375 | 376 | 377 | 378 | 379 | 380 | 381 | 382 | 383 | 384 | 385 | 386 | 387 | 388 | 389 | 390 | 391 | 392 | 393 | 394 | 395 | 396 | 397 | 398 | 399 | 400 | 401 | 402 | 403 | 404 | 405 | 406 | 407 | 408 | 409 | 410 | 411 | 412 | 413 | 414 | 415 | 416 | 417 | 418 | 419 | 420 | 421 | 422 | 423 | 424 | 425 | 426 | 427 | 428 | 429 | 430 | 431 | 432 | 433 | 434 | 435 | 436 | 437 | 438 | 439 | 440 | 441 | 442 | 443 | 444 | 445 | 446 | 447 | 448 | 449 | 450 | 451 | 452 | 453 | 454 | 455 | 456 | 457 | 458 | 459 | 460 | 461 | 462 | 463 | 464 | 465 | 466 | 467 | 468 | 469 | 470 | 471 | 472 | 473 | 474 | 475 | 476 | 477 | 478 | 479 | 480 | 481 | 482 | 483 | 484 | 485 | 486 | 487 | 488 | 489 | 490 | 491 | 492 | 493 | 494 | 495 | 496 | 497 | 498 | 499 | 500 | 501 | 502 | 503 | 504 | 505 | 506 | 507 | 508 | 509 | 510 | 511 | 512 | 513 | 514 | 515 | 516 | 517 | 518 | 519 | 520 | 521 | 522 | 523 | 524 | 525 | 526 | 527 | 528 | 529 | 530 | 531 | 532 | 533 | 534 | 535 | 536 | 537 | 538 | 539 | 540 | 541 | 542 | 543 | 544 | 545 | 546 | 547 | 548 | 549 | 550 | 551 | 552 | 553 | 554 | 555 | 556 | 557 | 558 | 559 | 560 | 561 | 562 | 563 | 564 | 565 | 566 | 567 | 568 | 569 | 570 | 571 | 572 | 573 | 574 | 575 | 576 | 577 | 578 | 579 | 580 | 581 | 582 | 583 | 584 | 585 | 586 | 587 | 588 | 589 | 589 | 590 | 591 | 592 | 593 | 594 | 595 | 596 | 597 | 598 | 599 | 600 | 601 | 602 | 603 | 604 | 605 | 606 | 607 | 608 | 609 | 610 | 611 | 612 | 613 | 614 | 615 | 616 | 617 | 618 | 619 | 619 | 620 | 621 | 622 | 623 | 624 | 625 | 626 | 627 | 628 | 629 | 629 | 630 | 631 | 632 | 633 | 634 | 635 | 636 | 637 | 638 | 639 | 639 | 640 | 641 | 642 | 643 | 644 | 645 | 646 | 647 | 648 | 649 | 649 | 650 | 651 | 652 | 653 | 654 | 655 | 656 | 657 | 658 | 659 | 659 | 660 | 661 | 662 | 663 | 664 | 665 | 666 | 667 | 668 | 669 | 669 | 670 | 671 | 672 | 673 | 674 | 675 | 676 | 677 | 678 | 679 | 679 | 680 | 681 | 682 | 683 | 684 | 685 | 686 | 687 | 688 | 689 | 689 | 690 | 691 | 692 | 693 | 694 | 695 | 696 | 697 | 698 | 699 | 699 | 700 | 701 | 702 | 703 | 704 | 705 | 706 | 707 | 708 | 709 | 709 | 710 | 711 | 712 | 713 | 714 | 715 | 716 | 717 | 718 | 719 | 719 | 720 | 721 | 722 | 723 | 724 | 725 | 726 | 727 | 728 | 729 | 729 | 730 | 731 | 732 | 733 | 734 | 735 | 736 | 737 | 738 | 739 | 739 | 740 | 741 | 742 | 743 | 744 | 745 | 746 | 747 | 748 | 749 | 749 | 750 | 751 | 752 | 753 | 754 | 755 | 756 | 757 | 758 | 759 | 759 | 760 | 761 | 762 | 763 | 764 | 765 | 766 | 767 | 768 | 769 | 769 | 770 | 771 | 772 | 773 | 774 | 775 | 776 | 777 | 778 | 779 | 779 | 780 | 781 | 782 | 783 | 784 | 785 | 786 | 787 | 788 | 789 | 789 | 790 | 791 | 792 | 793 | 794 | 795 | 796 | 797 | 798 | 799 | 799 | 800 | 801 | 802 | 803 | 804 | 805 | 806 | 807 | 808 | 809 | 809 | 810 | 811 | 812 | 813 | 814 | 815 | 816 | 817 | 818 | 819 | 819 | 820 | 821 | 822 | 823 | 824 | 825 | 826 | 827 | 828 | 829 | 829 | 830 | 831 | 832 | 833 | 834 | 835 | 836 | 837 | 838 | 839 | 839 | 840 | 841 | 842 | 843 | 844 | 845 | 846 | 847 | 848 | 849 | 849 | 850 | 851 | 852 | 853 | 854 | 855 | 856 | 857 | 858 | 859 | 859 | 860 | 861 | 862 | 863 | 864 | 865 | 866 | 867 | 868 | 869 | 869 | 870 | 871 | 872 | 873 | 874 | 875 | 876 | 877 | 878 | 879 | 879 | 880 | 881 | 882 | 883 | 884 | 885 | 886 | 887 | 888 | 889 | 889 | 890 | 891 | 892 | 893 | 894 | 895 | 896 | 897 | 898 | 899 | 899 | 900 | 901 | 902 | 903 | 904 | 905 | 906 | 907 | 908 | 909 | 909 | 910 | 911 | 912 | 913 | 914 | 915 | 916 | 917 | 918 | 919 | 919 | 920 | 921 | 922 | 923 | 924 | 925 | 926 | 927 | 928 | 929 | 929 | 930 | 931 | 932 | 933 | 934 | 935 | 936 | 937 | 938 | 939 | 939 | 940 | 941 | 942 | 943 | 944 | 945 | 946 | 947 | 948 | 949 | 949 | 950 | 951 | 952 | 953 | 954 | 955 | 956 | 957 | 958 | 959 | 959 | 960 | 961 | 962 | 963 | 964 | 965 | 966 | 967 | 968 | 969 | 969 | 970 | 971 | 972 | 973 | 974 | 975 | 976 | 977 | 978 | 979 | 979 | 980 | 981 | 982 | 983 | 984 | 985 | 986 | 987 | 988 | 989 | 989 | 990 | 991 | 992 | 993 | 994 | 995 | 996 | 997 | 998 | 999 | 999 | 1000 | 1001 | 1002 | 1003 | 1004 | 1005 | 1006 | 1007 | 1008 | 1009 | 1009 | 1010 | 1011 | 1012 | 1013 | 1014 | 1015 | 1016 | 1017 | 1018 | 1019 | 1019 | 1020 | 1021 | 1022 | 1023 | 1024 | 1025 | 1026 | 1027 | 1028 | 1029 | 1029 | 1030 | 1031 | 1032 | 1033 | 1034 | 1035 | 1036 | 1037 | 1038 | 1039 | 1039 | 1040 | 1041 | 1042 | 1043 | 1044 | 1045 | 1046 | 1047 | 1048 | 1049 | 1049 | 1050 | 1051 | 1052 | 1053 | 1054 | 1055 | 1056 | 1057 | 1058 | 1059 | 1059 | 1060 | 1061 | 1062 | 1063 | 1064 | 1065 | 1066 | 1067 | 1068 | 1069 | 1069 | 1070 | 1071 | 1072 | 1073 | 1074 | 1075 | 1076 | 1077 | 1078 | 1079 | 1079 | 1080 | 1081 | 1082 | 1083 | 1084 | 1085 | 1086 | 1087 | 1088 | 1089 | 1089 | 1090 | 1091 | 1092 | 1093 | 1094 | 1095 | 1096 | 1097 | 1098 | 1098 | 1099 | 1100 | 1101 | 1102 | 1103 | 1104 | 1105 | 1106 | 1107 | 1108 | 1109 | 1109 | 1110 | 1111 | 1112 | 1113 | 1114 | 1115 | 1116 | 1117 | 1118 | 1119 | 1119 | 1120 | 1121 | 1122 | 1123 | 1124 | 1125 | 1126 | 1127 | 1128 | 1129 | 1129 | 1130 | 1131 | 1132 | 1133 | 1134 | 1135 | 1136 | 1137 | 1138 | 1139 | 1139 | 1140 | 1141 | 1142 | 1143 | 1144 | 1145 | 1146 | 1147 | 1148 | 1149 | 1149 | 1150 | 1151 | 1152 | 1153 | 1154 | 1155 | 1156 | 1157 | 1158 | 1159 | 1159 | 1160 | 1161 | 1162 | 1163 | 1164 | 1165 | 1166 | 1167 | 1168 | 1169 | 1169 | 1170 | 1171 | 1172 | 1173 | 1174 | 1175 | 1176 | 1177 | 1178 | 1179 | 1179 | 1180 | 1181 | 1182 | 1183 | 1184 | 1185 | 1186 | 1187 | 1188 | 1189 | 1189 | 1190 | 1191 | 1192 | 1193 | 1194 | 1195 | 1196 | 1197 | 1198 | 1198 | 1199 | 1200 | 1201 | 1202 | 1203 | 1204 | 1205 | 1206 | 1207 | 1208 | 1209 | 1209 | 1210 | 1211 | 1212 | 1213 | 1214 | 1215 | 1216 | 1217 | 1218 | 1219 | 1219 | 1220 | 1221 | 1222 | 1223 | 1224 | 1225 | 1226 | 1227 | 1228 | 1229 | 1229 | 1230 | 1231 | 1232 | 1233 | 1234 | 1235 | 1236 | 1237 | 1238 | 1239 | 1239 | 1240 | 1241 | 1242 | 1243 | 1244 | 1245 | 1246 | 1247 | 1248 | 1249 | 1249 | 1250 | 1251 | 1252 | 1253 | 1254 | 1255 | 1256 | 1257 | 1258 | 1259 | 1259 | 1260 | 1261 | 1262 | 1263 | 1264 | 1265 | 1266 | 1267 | 1268 | 1269 | 1269 | 1270 | 1271 | 1272 | 1273 | 1274 | 1275 | 1276 | 1277 | 1278 | 1279 | 1279 | 1280 | 1281 | 1282 | 1283 | 1284 | 1285 | 1286 | 1287 | 1288 | 1289 | 1289 | 1290 | 1291 | 1292 | 1293 | 1294 | 1295 | 1296 | 1297 | 1298 | 1298 | 1299 | 1300 | 1301 | 1302 | 1303 | 1304 | 1305 | 1306 | 1307 | 1308 | 1309 | 1309 | 1310 | 1311 | 1312 | 1313 | 1314 | 1315 | 1316 | 1317 | 1318 | 1319 | 1319 | 1320 | 1321 | 1322 | 1323 | 1324 | 1325 | 1326 | 1327 | 1328 | 1329 | 1329 | 1330 | 1331 | 1332 | 1333 | 1334 | 1335 | 1336 | 1337 | 1338 | 1339 | 1339 | 1340 | 1341 | 1342 | 1343 | 1344 | 1345 | 1346 | 1347 | 1348 | 1349 | 1349 | 1350 | 1351 | 1352 | 1353 | 1354 | 1355 | 1356 | 1357 | 1358 | 1359 | 1359 | 1360 | 1361 | 1362 | 1363 | 1364 | 1365 | 1366 | 1367 | 1368 | 1369 | 1369 | 1370 | 1371 | 1372 | 1373 | 1374 | 1375 | 1376 | 1377 | 1378 | 1379 | 1379 | 1380 | 1381 | 1382 | 1383 | 1384 | 1385 | 1386 | 1387 | 1388 | 1389 | 1389 | 1390 | 1391 | 1392 | 1393 | 1394 | 1395 | 1396 | 1397 | 1398 | 1398 | 1399 | 1400 | 1401 | 1402 | 1403 | 1404 | 1405 | 1406 | 1407 | 1408 | 1409 | 1409 | 1410 | 1411 | 1412 | 1413 | 1414 | 1415 | 1416 | 1417 | 1418 | 1419 | 1419 | 1420 | 1421 | 1422 | 1423 | 1424 | 1425 | 1426 | 1427 | 1428 | 1429 | 1429 | 1430 | 1431 | 1432 | 1433 | 1434 | 1435 | 1436 | 1437 | 1438 | 1439 | 1439 | 1440 | 1441 | 1442 | 1443 | 1444 | 1445 | 1446 | 1447 | 1448 | 1449 | 1449 | 1450 | 1451 | 1452 | 1453 | 1454 | 1455 | 1456 | 1457 | 1458 | 1459 | 1459 | 1460 | 1461 | 1462 | 1463 | 1464 | 1465 | 1466 | 1467 | 1468 | 1469 | 1469 | 1470 | 1471 | 1472 | 1473 | 1474 | 1475 | 1476 | 1477 | 1478 | 1479 | 1479 | 1480 | 1481 | 1482 | 1483 | 1484 | 1485 | 1486 | 1487 | 1488 | 1489 | 1489 | 1490 | 1491 | 1492 | 1493 | 1494 | 1495 | 1496 | 1497 | 1498 | 1498 | 1499 | 1500 | 1501 | 1502 | 1503 | 1504 | 1505 | 1506 | 1507 | 1508 | 1509 | 1509 | 1510 | 1511 | 1512 | 1513 | 1514 | 1515 | 1516 | 1517 | 1518 | 1519 | 1519 | 1520 | 1521 | 1522 | 1523 | 1524 | 1525 | 1526 | 1527 | 1528 | 1529 | 1529 | 1530 | 1531 | 1532 | 1533 | 1534 | 1535 | 1536 | 1537 | 1538 | 1539 | 1539 | 1540 | 1541 | 1542 | 1543 | 1544 | 1545 | 1546 | 1547 | 1548 | 1549 | 1549 | 1550 | 1551 | 1552 | 1553 | 1554 | 1555 | 1556 | 1557 | 1558 | 1559 | 1559 | 1560 | 1561 | 1562 | 1563 | 1564 | 1565 | 1566 | 1567 | 1568 | 1569 | 1569 | 1570 | 1571 | 1572 | 1573 | 1574 | 1575 | 1576 | 1577 | 1578 | 1579 | 1579 | 1580 | 1581 | 1582 | 1583 | 1584 | 1585 | 1586 | 1587 | 1588 | 1589 | 1589 | 1590 | 1591 | 1592 | 1593 | 1594 | 1595 | 1596 | 1597 | 1598 | 1598 | 1599 | 1600 | 1601 | 1602 | 1603 | 1604 | 1605 | 1606 | 1607 | 1608 | 1609 | 1609 | 1610 | 1611 | 1612 | 1613 | 1614 | 1615 | 1616 | 1617 | 1618 | 1619 | 1619 | 1620 | 1621 | 1622 | 1623 | 1624 | 1625 | 1626 | 1627 | 1628 | 1629 | 1629 | 1630 | 1631 | 1632 | 1633 | 1634 | 1635 | 1636 | 1637 | 1638 | 1639 | 1639 | 1640 | 1641 | 1642 | 1643 | 1644 | 1645 | 1646 | 1647 | 1648 | 1649 | 1649 | 1650 | 1651 | 1652 | 1653 | 1654 | 1655 | 1656 | 1657 | 1658 | 1659 | 1659 | 1660 | 1661 | 1662 | 1663 | 1664 | 1665 | 1666 | 1667 | 1668 | 1669 | 1669 | 1670 | 1671 | 1672 | 1673 | 1674 | 1675 | 1676 | 1677 | 1678 | 1679 | 1679 | 1680 | 1681 | 1682 | 1683 | 1684 | 1685 | 1686 | 1687 | 1688 | 1689 | 1689 | 1690 | 1691 | 1692 | 1693 | 1694 | 1695 | 1696 | 1697 | 1698 | 1698 | 1699 | 1700 | 1701 | 1702 | 1703 | 1704 | 1705 | 1706 | 1707 | 1708 | 1709 | 1709 | 1710 | 1711 | 1712 | 1713 | 1714 | 1715 | 1716 | 1717 | 1718 | 1719 | 1719 | 1720 | 1721 | 1722 | 1723 | 1724 | 1725 | 1726 | 1727 | 1728 | 1729 | 1729 | 1730 | 1731 | 1732 | 1733 | 1734 | 1735 | 1736 | 1737 | 1738 | 1739 | 1739 | 1740 | 1741 | 1742 | 1743 | 1744 | 1745 | 1746 | 1747 | 1748 | 1749 | 1749 | 1750 | 1751 | 1752 | 1753 | 1754 | 1755 | 1756 | 1757 | 1758 | 1759 | 1759 | 1760 | 1761 | 1762 | 1763 | 1764 | 1765 | 1766 | 1767 | 1768 | 1769 | 1769 | 1770 | 1771 | 1772 | 1773 | 1774 | 1775 | 1776 | 1777 | 1778 | 1779 | 1779 | 1780 | 1781 | 1782 | 178
```

Testpath

4 test paths are needed for Edge Coverage

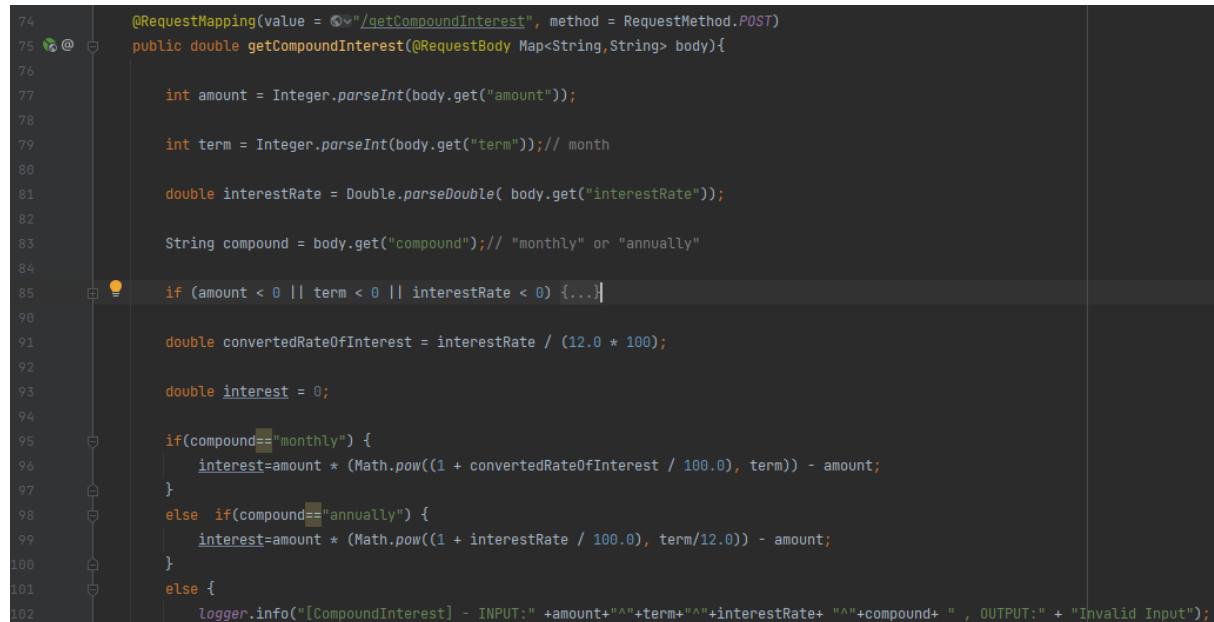
[1,2,3,4]
[1,2,5,6,8,10,11]
[1,2,5,6,7,3,4]
[1,2,5,6,7,9,10,11]

4 test paths are needed for Prime Path Coverage

Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,7,9,10,11]	[1,2,5,6,7,9,10,11]
[1,2,5,6,7,3,4]	[1,2,5,6,7,3,4]
[1,2,5,6,8,10,11]	[1,2,5,6,8,10,11]
[1,2,3,4]	[1,2,3,4]

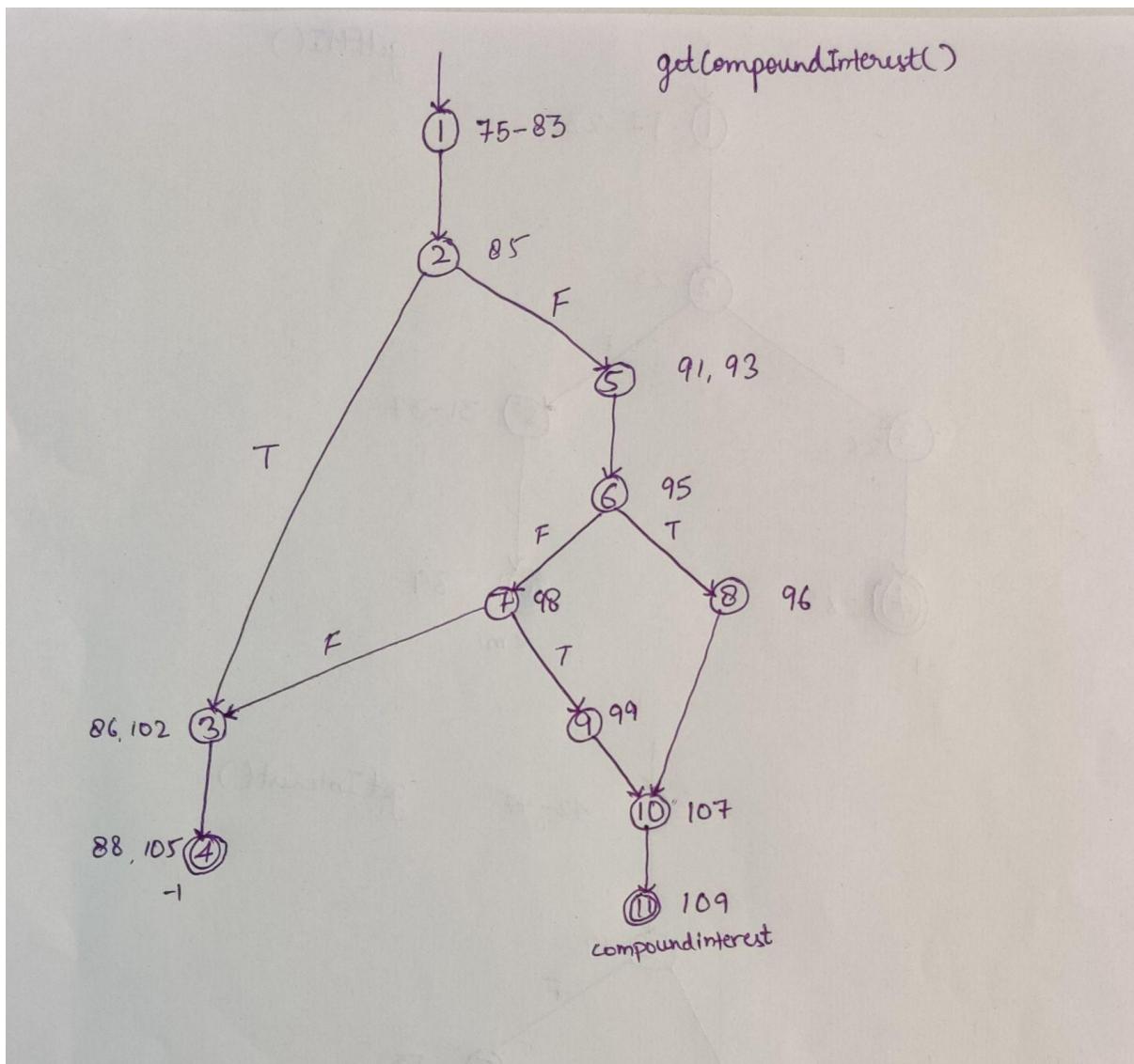
3. getCompoundInterest

- This function is used to get the Compound Interest



```
74     @RequestMapping(value = "/getCompoundInterest", method = RequestMethod.POST)
75     public double getCompoundInterest(@RequestBody Map<String, String> body){
76
77         int amount = Integer.parseInt(body.get("amount"));
78
79         int term = Integer.parseInt(body.get("term")); // month
80
81         double interestRate = Double.parseDouble(body.get("interestRate"));
82
83         String compound = body.get("compound"); // "monthly" or "annually"
84
85         if (amount < 0 || term < 0 || interestRate < 0) {...}
86
87         double convertedRateOfInterest = interestRate / (12.0 * 100);
88
89         double interest = 0;
90
91         if(compound=="monthly") {
92             interest=amount * (Math.pow((1 + convertedRateOfInterest / 100.0), term)) - amount;
93         }
94         else if(compound=="annually") {
95             interest=amount * (Math.pow((1 + interestRate / 100.0), term/12.0)) - amount;
96         }
97         else {
98             logger.info("[CompoundInterest] - INPUT:" +amount+ "+"+term+ "+"+interestRate+ "+"+compound+ " , OUTPUT:" + "Invalid Input");
99         }
100    }
```

CFG and Block Number



Testcase

```

Run: BankTest.CompoundInterest X
  ✓ Tests passed: 1 of 1 test - 1 sec 234 ms
BankTest (com.example.Softw 1 sec 234 ms)
  ✓ CompoundInterest 1 sec 234 ms
    C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
    26/Nov/2022:15:30:21 +0530 [BankController.java] [INFO] BankController [CompoundInterest] - INPUT:1000^-1^10.0^monthly , 0
    26/Nov/2022:15:30:21 +0530 [BankController.java] [INFO] BankController [CompoundInterest] - INPUT:1000^5^10.0^q , OUTPUT:I
    26/Nov/2022:15:30:21 +0530 [BankController.java] [INFO] BankController [CompoundInterest] - INPUT:1000^15^10.0^monthly , 0
    26/Nov/2022:15:30:21 +0530 [BankController.java] [INFO] BankController [CompoundInterest] - INPUT:1000^25^10.0^annually ,
Process finished with exit code 0
  
```

Testpath

4 test paths are needed for Edge Coverage
[1,2,3,4]
[1,2,5,6,8,10,11]
[1,2,5,6,7,3,4]
[1,2,5,6,7,9,10,11]

4 test paths are needed for Prime Path Coverage

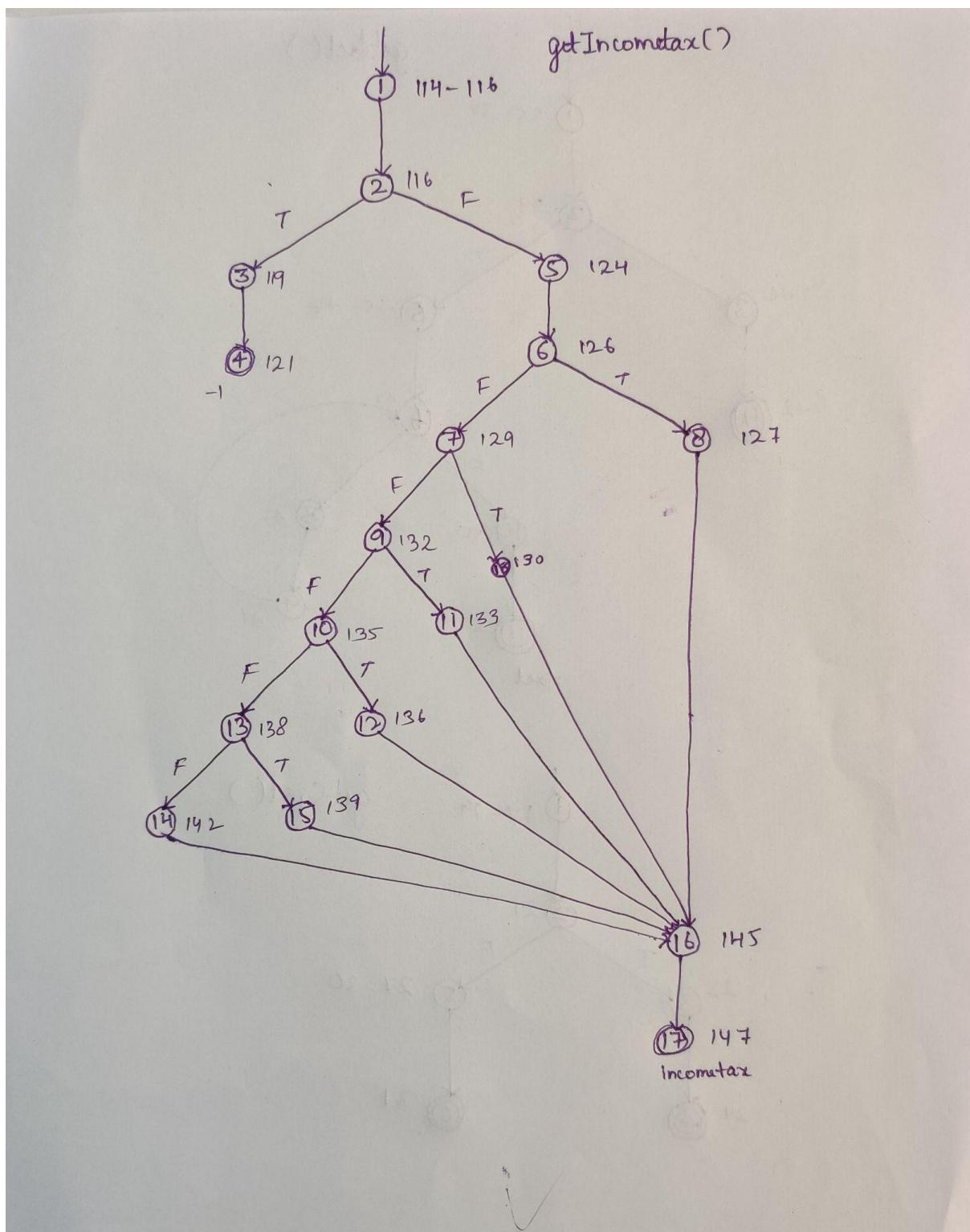
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,7,9,10,11]	[1,2,5,6,7,9,10,11]
[1,2,5,6,7,3,4]	[1,2,5,6,7,3,4]
[1,2,5,6,8,10,11]	[1,2,5,6,8,10,11]
[1,2,3,4]	[1,2,3,4]

4. getIncomeTax

- This function is used to get the Income tax.

```
114 @POST /income-tax  
115     public double getIncomeTax(@RequestBody Map<String, String> body){  
116         double income = Double.parseDouble(body.get("income"));  
117         if (income < 0) {...}  
118         double incomeTax=0;  
119         if (income <= 500000) {  
120             incomeTax = 0;  
121         }  
122         else if (income <= 750000) {  
123             incomeTax = (10.0/100 * (income - 500000)) ;  
124         }  
125         else if (income <= 1000000) {  
126             incomeTax = (15.0/100 * (income - 750000)) + ((750000 - 500000) * 10.0/100) ;  
127         }  
128         else if (income <= 1250000) {  
129             incomeTax = ((income - 1000000) * 20/100) + (15.0/100 * (1000000 - 750000)) + ((750000 - 500000) * 10.0/100) ;  
130         }  
131         else if (income <= 1500000) {  
132             incomeTax = ((income - 1250000) * 25/100) + (20.0/100 * (1250000 - 1000000)) + (15.0/100 * (1000000 - 750000)) + ((750000 - 500000) * 10.0/100) ;  
133         }  
134         else {  
135             incomeTax = ((income - 2000000) * 30/100) + (25.0/100 * (2000000 - 1250000)) +(20.0/100 * (1250000 - 1000000)) + (15.0/100 * (1000000 - 750000)) ;  
136         }  
137     }
```

CFG and Block Number



Testcase

```
Run: BankTest.IncomeTax <img alt="Run icon" data-bbox="60 178 75 195"/> Tests passed: 1 of 1 test – 1 sec 172 ms  
BankTest (com.example.Softw, 1 sec 172 ms)  
  IncomeTax 1 sec 172 ms  
C:\Users\drada\.jdks\openjdk-18.0.1\bin\java.exe ...  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:-10000.0 , OUTPUT:Invalid Input  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:40000.0 , OUTPUT:0.0  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:70000.0 , OUTPUT:20000.0  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:90000.0 , OUTPUT:47500.0  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:110000.0 , OUTPUT:82500.0  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:140000.0 , OUTPUT:150000.0  
26/Nov/2022:15:30:52 +0530 [BankController.java] [INFO] BankController [IncomeTax] - INPUT:160000.0 , OUTPUT:180000.0  
  
Process finished with exit code 0
```

Testpath

7 test paths are needed for Edge Coverage

[1,2,3,4]

[1,2,5,6,8,16,17]

[1,2,5,6,7,18,16,17]

[1,2,5,6,7,9,11,16,17]

[1,2,5,6,7,9,10,12,16,17]

[1,2,5,6,7,9,10,13,14,16,17]

[1,2,5,6,7,9,10,13,15,16,17]

7 test paths are needed for Prime Path Coverage

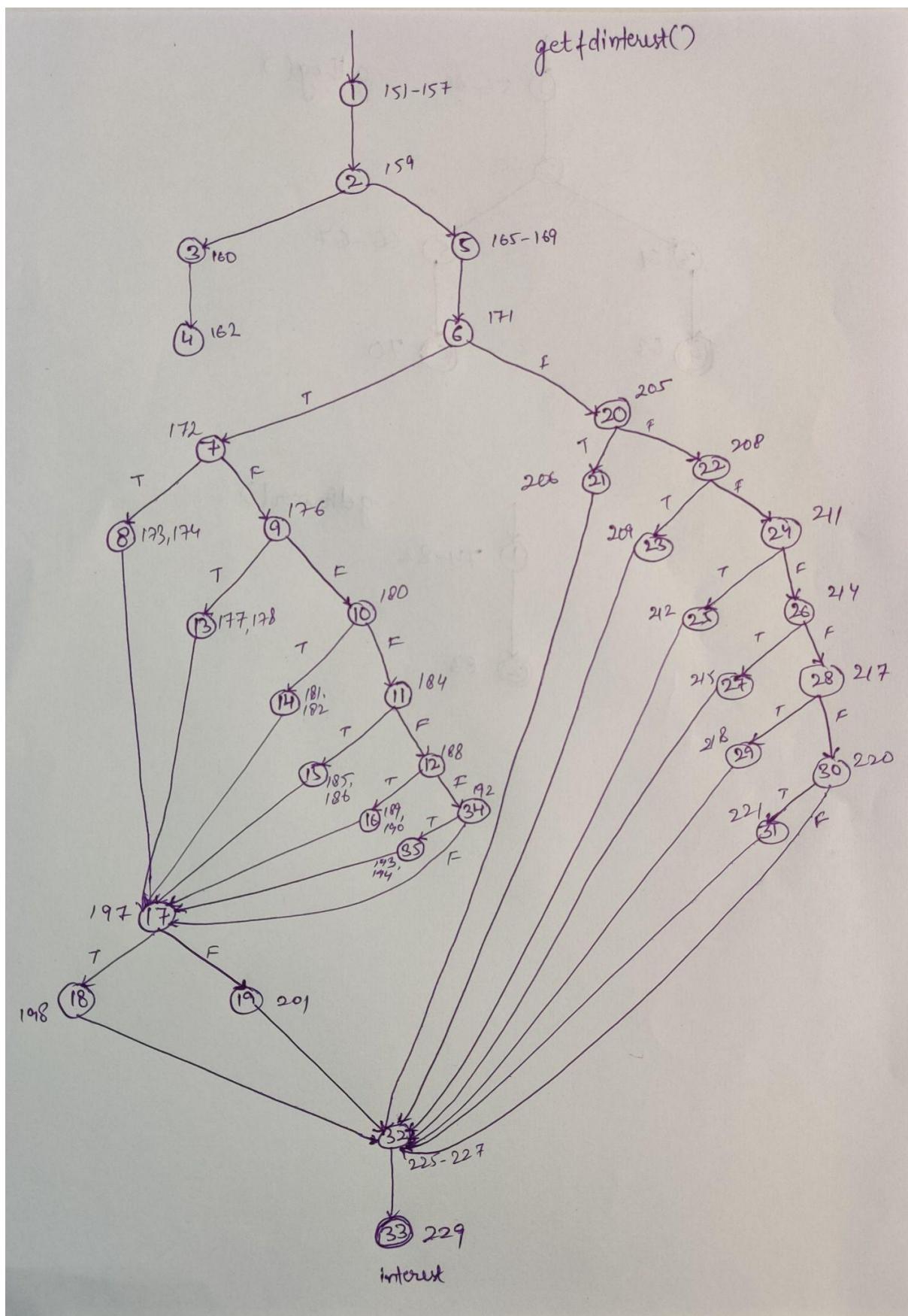
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,7,9,10,13,15,16,17]	[1,2,5,6,7,9,10,13,15,16,17]
[1,2,5,6,7,9,10,13,14,16,17]	[1,2,5,6,7,9,10,13,14,16,17]
[1,2,5,6,7,9,10,12,16,17]	[1,2,5,6,7,9,10,12,16,17]
[1,2,5,6,7,9,11,16,17]	[1,2,5,6,7,9,11,16,17]
[1,2,5,6,7,18,16,17]	[1,2,5,6,7,18,16,17]
[1,2,5,6,8,16,17]	[1,2,5,6,8,16,17]
[1,2,3,4]	[1,2,3,4]

5. getFDInterest

- This function is used to get the FD interest.

```
150 @RequestMapping(value = "/getFDInterest", method = RequestMethod.POST)
151 public double getFDInterest(@RequestBody Map<String, String> body){
152
153     int amount = Integer.parseInt(body.get("amount"));
154
155     int term = Integer.parseInt(body.get("term")); // days
156
157     int age = Integer.parseInt(body.get("age")); // years
158
159     if (amount < 0 || term < 0 || age < 0) {...}
160
161     double interestRate=0.0;
162
163     double interest = 0.0;
164
165     double interestRate1=0.0, interestRate2=0.0;
166
167
168     if (amount < 10000000) {
169
170         if (term >= 7 && term <= 14) {...}
171         else if (term >= 15 && term <= 29) {...}
172         else if (term >= 30 && term <= 45) {...}
173         else if (term >= 45 && term <= 60) {...}
174         else if (term >= 61 && term <= 184) {...}
175         else if (term >= 185 && term <= 365) {...}
176
177         if(age < 50) {...}
178         else {...}
179
180     }
181
182 }
```

CFG and Block Number



Testcase

```
Run: BankTest.FDInterest
BankTest (com.example.Softw, 1 sec 599 ms) Tests passed: 1 of 1 test – 1 sec 599 ms
  ✓ FDInterest 1 sec 599 ms
    C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^1^10 , OUTPUT:Invalid
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^12^10 , OUTPUT:45000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^24^10 , OUTPUT:47000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^36^10 , OUTPUT:55000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^48^10 , OUTPUT:70000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^120^10 , OUTPUT:75000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^240^10 , OUTPUT:80000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^12^60 , OUTPUT:50000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^24^60 , OUTPUT:52500.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^36^60 , OUTPUT:60000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^48^60 , OUTPUT:75000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^120^60 , OUTPUT:80000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^240^60 , OUTPUT:85000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^12^60 , OUTPUT:65000.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^24^60 , OUTPUT:75000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^36^60 , OUTPUT:67500.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^48^60 , OUTPUT:80000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^120^60 , OUTPUT:85000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^240^60 , OUTPUT:100000.
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:1000000^480^10 , OUTPUT:0.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:10000000^480^60 , OUTPUT:0.0
    26/Nov/2022:15:23:58 +0530 [BankController.java] [INFO] BankController [FDInterest] - INPUT:10000000^480^50 , OUTPUT:0.0
  Activate Windows
```

Testpath

16 test paths are needed for Edge Coverage

- [1,2,3,4]
- [1,2,5,6,20,21,32,33]
- [1,2,5,6,20,22,23,32,33]
- [1,2,5,6,20,22,24,25,32,33]
- [1,2,5,6,7,8,17,19,32,33]
- [1,2,5,6,7,8,17,18,32,33]
- [1,2,5,6,20,22,24,26,27,32,33]
- [1,2,5,6,20,22,24,26,28,30,32,33]
- [1,2,5,6,20,22,24,26,28,29,32,33]
- [1,2,5,6,20,22,24,26,28,30,31,32,33]
- [1,2,5,6,7,9,13,17,19,32,33]
- [1,2,5,6,7,9,10,14,17,19,32,33]
- [1,2,5,6,7,9,10,11,15,17,19,32,33]
- [1,2,5,6,7,9,10,11,12,34,17,19,32,33]
- [1,2,5,6,7,9,10,11,12,16,17,19,32,33]
- [1,2,5,6,7,9,10,11,12,34,35,17,19,32,33]

22 test paths are needed for Prime Path Coverage

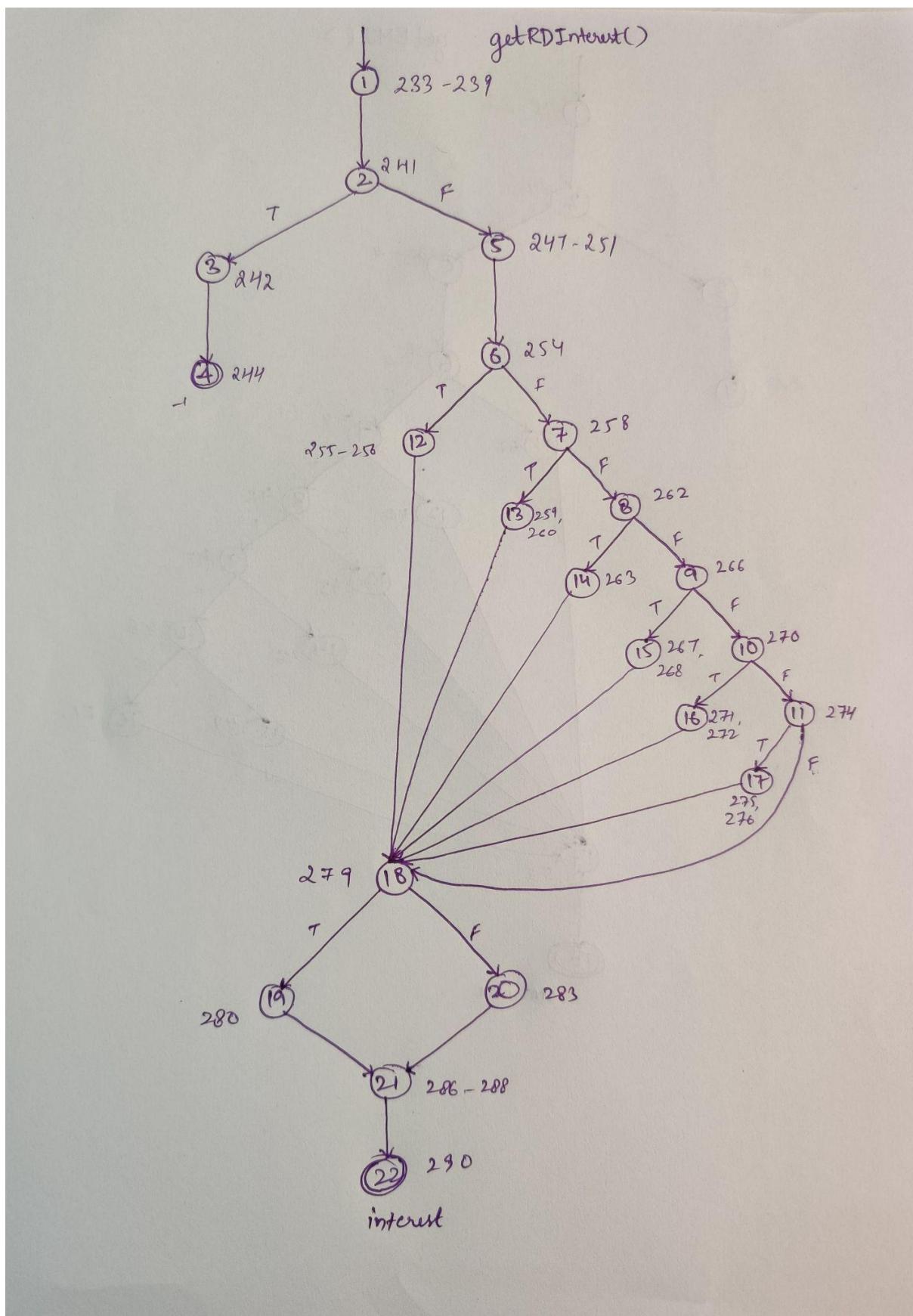
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,7,9,10,11,12,34,35,17,18,32,33]	[1,2,5,6,7,9,10,11,12,34,35,17,18,32,33]
[1,2,5,6,7,9,10,11,12,34,35,17,19,32,33]	[1,2,5,6,7,9,10,11,12,34,35,17,19,32,33]
[1,2,5,6,7,9,10,11,12,16,17,19,32,33]	[1,2,5,6,7,9,10,11,12,16,17,19,32,33]
[1,2,5,6,7,9,10,11,12,16,17,18,32,33]	[1,2,5,6,7,9,10,11,12,16,17,18,32,33]
[1,2,5,6,7,9,10,11,12,34,17,18,32,33]	[1,2,5,6,7,9,10,11,12,34,17,18,32,33]
[1,2,5,6,7,9,10,11,12,34,17,19,32,33]	[1,2,5,6,7,9,10,11,12,34,17,19,32,33]
[1,2,5,6,7,9,10,11,15,17,18,32,33]	[1,2,5,6,7,9,10,11,15,17,18,32,33]
[1,2,5,6,7,9,10,11,15,17,19,32,33]	[1,2,5,6,7,9,10,11,15,17,19,32,33]
[1,2,5,6,20,22,24,26,28,30,31,32,33]	[1,2,5,6,20,22,24,26,28,30,31,32,33]
[1,2,5,6,20,22,24,26,28,29,32,33]	[1,2,5,6,20,22,24,26,28,29,32,33]
[1,2,5,6,20,22,24,26,28,30,32,33]	[1,2,5,6,20,22,24,26,28,30,32,33]
[1,2,5,6,7,9,10,14,17,19,32,33]	[1,2,5,6,7,9,10,14,17,19,32,33]
[1,2,5,6,7,9,10,14,17,18,32,33]	[1,2,5,6,7,9,10,14,17,18,32,33]
[1,2,5,6,7,9,13,17,18,32,33]	[1,2,5,6,7,9,13,17,18,32,33]
[1,2,5,6,7,9,13,17,19,32,33]	[1,2,5,6,7,9,13,17,19,32,33]
[1,2,5,6,20,22,24,26,27,32,33]	[1,2,5,6,20,22,24,26,27,32,33]
[1,2,5,6,20,22,24,25,32,33]	[1,2,5,6,20,22,24,25,32,33]
[1,2,5,6,7,8,17,18,32,33]	[1,2,5,6,7,8,17,18,32,33]
[1,2,5,6,7,8,17,19,32,33]	[1,2,5,6,7,8,17,19,32,33]
[1,2,5,6,20,22,23,32,33]	[1,2,5,6,20,22,23,32,33]
[1,2,5,6,20,21,32,33]	[1,2,5,6,20,21,32,33]
[1,2,3,4]	[1,2,3,4]

6. getRDInterest

- This function is used to get the RD interest.

```
233  @ 
234      public double getRDInterest(@RequestBody Map<String, String> body){
235          int amount = Integer.parseInt(body.get("amount"));
236
237          int term = Integer.parseInt(body.get("term")); // months
238
239          int age = Integer.parseInt(body.get("age")); // years
240
241          if (amount < 0 || term < 0 || age < 0) {...}
242
243          double interestRate=0.0;
244
245          double interest = 0.0;
246
247          double interestRate1=0.0, interestRate2=0.0;
248
249
250
251
252
253
254          if (term >= 0 && term <= 6) {...}
255          else if (term >= 7 && term <= 9) {...}
256          else if (term >= 10 && term <= 12) {...}
257          else if (term >= 13 && term <= 15) {...}
258          else if (term >= 16 && term <= 18) {...}
259          else if (term <= 22) {...}
260
261
262          if(age < 50) {...}
263          else[...]
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
```

CFG and Block Number



Testcase

```

Run: BankTest.RDInterest
BankTest (com.example.Softw. 1 sec 214 ms)
  ✓ RDInterest 1 sec 214 ms
    C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^1^10 , OUTPUT:Invalid
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^6^10 , OUTPUT:75000.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^9^10 , OUTPUT:77500.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^12^10 , OUTPUT:80000.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^15^10 , OUTPUT:82500.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^18^10 , OUTPUT:85000.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^21^10 , OUTPUT:87500.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^24^10 , OUTPUT:0.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^6^60 , OUTPUT:80000.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^9^60 , OUTPUT:82500.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^12^60 , OUTPUT:85000.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^15^60 , OUTPUT:87500.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^18^60 , OUTPUT:90000.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^21^60 , OUTPUT:92500.0
    26/Nov/2022:15:26:13 +0530 [BankController.java] [INFO] BankController [RDInterest] - INPUT:1000000^24^60 , OUTPUT:0.0

Process finished with exit code 0

```

Testpath

9 test paths are needed for Edge Coverage

- [1,2,3,4]
- [1,2,5,6,12,18,20,21,22]
- [1,2,5,6,7,13,18,20,21,22]
- [1,2,5,6,12,18,19,21,22]
- [1,2,5,6,7,8,14,18,20,21,22]
- [1,2,5,6,7,8,9,15,18,20,21,22]
- [1,2,5,6,7,8,9,10,11,18,20,21,22]
- [1,2,5,6,7,8,9,10,16,18,20,21,22]
- [1,2,5,6,7,8,9,10,11,17,18,20,21,22]

15 test paths are needed for Prime Path Coverage

Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,7,8,9,10,11,17,18,19,21,22]	[1,2,5,6,7,8,9,10,11,17,18,19,21,22]
[1,2,5,6,7,8,9,10,11,17,18,20,21,22]	[1,2,5,6,7,8,9,10,11,17,18,20,21,22]
[1,2,5,6,7,8,9,10,16,18,20,21,22]	[1,2,5,6,7,8,9,10,16,18,20,21,22]
[1,2,5,6,7,8,9,10,16,18,19,21,22]	[1,2,5,6,7,8,9,10,16,18,19,21,22]
[1,2,5,6,7,8,9,10,11,18,19,21,22]	[1,2,5,6,7,8,9,10,11,18,19,21,22]
[1,2,5,6,7,8,9,10,11,18,20,21,22]	[1,2,5,6,7,8,9,10,11,18,20,21,22]
[1,2,5,6,7,8,9,15,18,19,21,22]	[1,2,5,6,7,8,9,15,18,19,21,22]
[1,2,5,6,7,8,9,15,18,20,21,22]	[1,2,5,6,7,8,9,15,18,20,21,22]
[1,2,5,6,7,8,14,18,20,21,22]	[1,2,5,6,7,8,14,18,20,21,22]
[1,2,5,6,7,8,14,18,19,21,22]	[1,2,5,6,7,8,14,18,19,21,22]
[1,2,5,6,7,13,18,20,21,22]	[1,2,5,6,7,13,18,20,21,22]
[1,2,5,6,7,13,18,19,21,22]	[1,2,5,6,7,13,18,19,21,22]
[1,2,5,6,12,18,20,21,22]	[1,2,5,6,12,18,20,21,22]
[1,2,5,6,12,18,19,21,22]	[1,2,5,6,12,18,19,21,22]
[1,2,3,4]	[1,2,3,4]

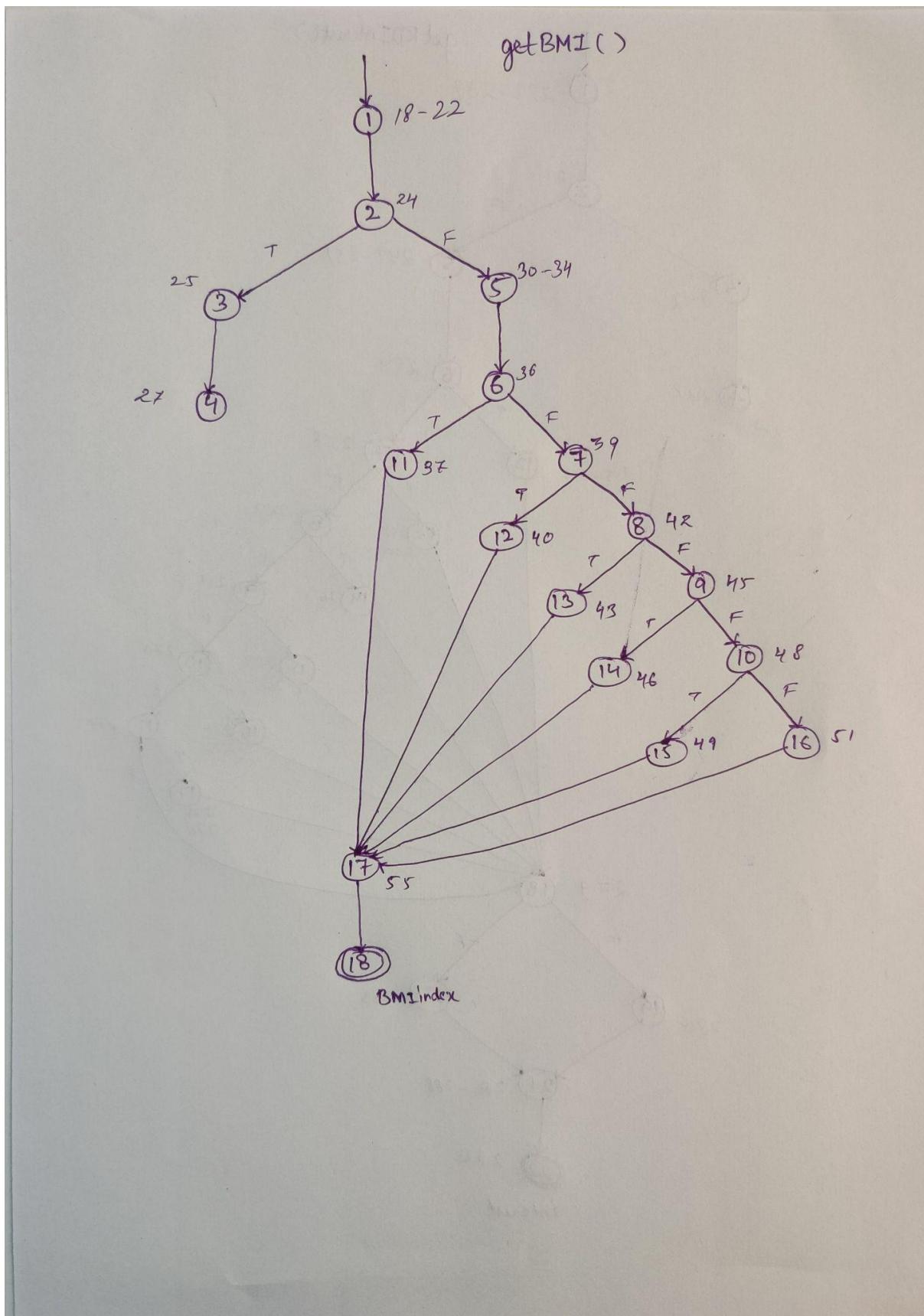
II. Health Calculator

1. getBMI

- This function is used to calculate the Body Mass Index.

```
18  public String getBMI(@RequestBody Map<String, String> body){  
19  
20      double height = Double.parseDouble( body.get("height")); //cms  
21  
22      double weight = Double.parseDouble( body.get("weight")); //kgs  
23  
24      if (height < 0 || weight < 0) {...}  
25  
26      height = height / 100;  
27  
28      double bmi = weight / (height * height);  
29  
30  
31      String bmiIndex;  
32  
33  
34      if (bmi < 16) {...}  
35      else if (bmi < 16.99999999 && bmi >= 16) {...}  
36      else if (bmi < 18.4 && bmi >= 17) {...}  
37      else if (bmi < 25 && bmi >= 18.4) {...}  
38      else if (bmi < 29.4 && bmi >= 25) {...}  
39      else{...}  
40  
41      logger.info("[BMI] - INPUT:" +height+"^"+weight+ " , OUTPUT:" + bmi+"^"+bmiIndex);  
42  
43      return bmiIndex ;  
44  }
```

CFG and Block Number



Testcase

```
Run: 4 HealthTest.BMI
C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
Tests passed: 1 of 1 test - 1 sec 359 ms
HealthTest (com.example.Softb 1 sec 359 ms)
  ✓ BMI 1 sec 359 ms
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:180.0^1.0 , OUTPUT:Invalid Input
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:1.8^40.0 , OUTPUT:12.34567901234
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:1.8^55.0 , OUTPUT:16.97530864197
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:1.8^58.0 , OUTPUT:17.90123456790
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:1.8^65.0 , OUTPUT:20.06172839506
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:1.8^85.0 , OUTPUT:26.234567901234
    26/Nov/2022:15:34:29 +0530 [HealthController.java] [INFO] HealthController [BMI] - INPUT:1.8^110.0 , OUTPUT:33.9506172839

Process finished with exit code 0
```

Testpath

7 test paths are needed for Edge Coverage

[1,2,3,4]
[1,2,5,6,11,17,18]
[1,2,5,6,7,12,17,18]
[1,2,5,6,7,8,13,17,18]
[1,2,5,6,7,8,9,14,17,18]
[1,2,5,6,7,8,9,10,16,17,18]
[1,2,5,6,7,8,9,10,15,17,18]

7 test paths are needed for Prime Path Coverage

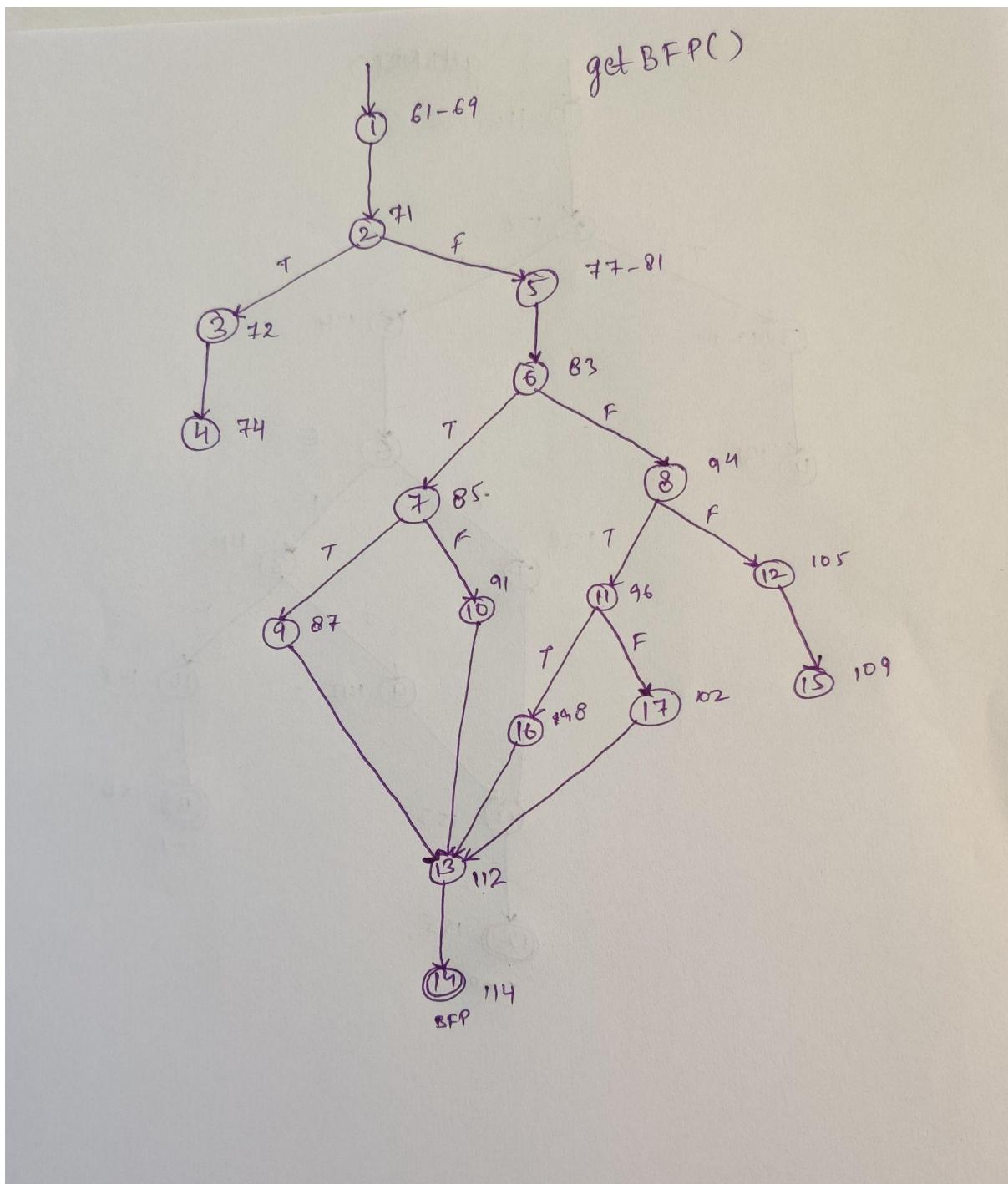
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,7,8,9,10,15,17,18]	[1,2,5,6,7,8,9,10,15,17,18]
[1,2,5,6,7,8,9,10,16,17,18]	[1,2,5,6,7,8,9,10,16,17,18]
[1,2,5,6,7,8,9,14,17,18]	[1,2,5,6,7,8,9,14,17,18]
[1,2,5,6,7,8,13,17,18]	[1,2,5,6,7,8,13,17,18]
[1,2,5,6,7,12,17,18]	[1,2,5,6,7,12,17,18]
[1,2,5,6,11,17,18]	[1,2,5,6,11,17,18]
[1,2,3,4]	[1,2,3,4]

2. getBFP

- This function is used to calculate the Body Fat Percentage.

```
61  @ public double getBFP(@RequestBody Map<String, String> body){  
62  
63      double height = Double.parseDouble( body.get("height")); //cms  
64  
65      double weight = Double.parseDouble( body.get("weight")); //kgs  
66  
67      int age = Integer.parseInt(body.get("age")); // years  
68  
69      String gender = body.get("gender"); // male , female  
70  
71      if (height < 0 || weight < 0 || age<0 ) {...}  
72  
73  
74      height = height / 100;  
75  
76  
77      double bmi = weight / (height * height);  
78  
79  
80      double bfp=0.0;  
81  
82  
83      if(gender=="male")  
84      {  
85          if(age<=14)  
86          {...}  
87          else  
88          {...}  
89      }  
90      else if(gender=="female")  
91      {  
92          if(age<=14)  
93          {...}  
94          else  
95          {...}  
96      }
```

CFG and Block Number



Testcase

Screenshot of a Java IDE showing the results of a test run. The test passed with 1 of 1 test in 1 sec 324 ms.

```
Run: HealthTest.BFP ×
  ✓ Tests passed: 1 of 1 test – 1 sec 324 ms
  ✓ HealthTest (com.example.Soft 1 sec 324 ms
    ✓ BFP 1 sec 324 ms
      C:\Users\drada\jdk8\openjdk-18.0.1.1\bin\java.exe ...
      26/Nov/2022:15:34:57 +0530 [HealthController.java] [INFO] HealthController [BFP] - INPUT:180.0^-1.0^18^male, OUTPUT:Invalid Input
      26/Nov/2022:15:34:57 +0530 [HealthController.java] [INFO] HealthController [BFP] - INPUT:1.8^40.0^18^ma, OUTPUT:Invalid Input
      26/Nov/2022:15:34:57 +0530 [HealthController.java] [INFO] HealthController [BFP] - INPUT:1.8^55.0^18^male, OUTPUT:36.03271
      26/Nov/2022:15:34:57 +0530 [HealthController.java] [INFO] HealthController [BFP] - INPUT:1.8^60.0^10^male, OUTPUT:8.322222
      26/Nov/2022:15:34:57 +0530 [HealthController.java] [INFO] HealthController [BFP] - INPUT:1.8^65.0^18^female, OUTPUT:41.493
      26/Nov/2022:15:34:57 +0530 [HealthController.java] [INFO] HealthController [BFP] - INPUT:1.8^85.0^10^female, OUTPUT:28.381
      Process finished with exit code 0
```

Testpath

6 test paths are needed for Edge Coverage

- [1,2,3,4]
- [1,2,5,6,8,12,15]
- [1,2,5,6,7,9,13,14]
- [1,2,5,6,7,10,13,14]
- [1,2,5,6,8,11,17,13,14]
- [1,2,5,6,8,11,16,13,14]

6 test paths are needed for Prime Path Coverage

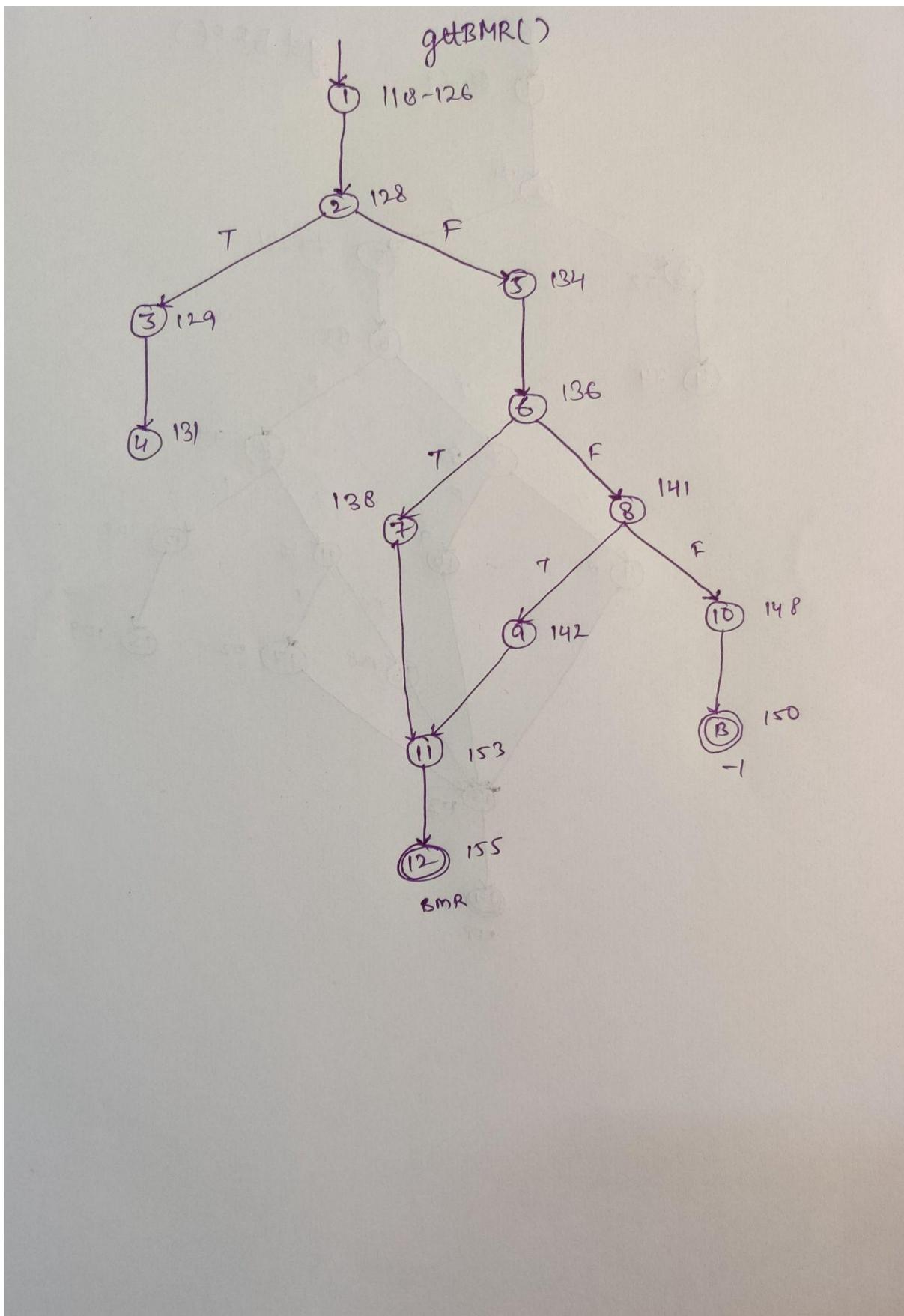
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,8,11,16,13,14]	[1,2,5,6,8,11,16,13,14]
[1,2,5,6,8,11,17,13,14]	[1,2,5,6,8,11,17,13,14]
[1,2,5,6,7,9,13,14]	[1,2,5,6,7,9,13,14]
[1,2,5,6,7,10,13,14]	[1,2,5,6,7,10,13,14]
[1,2,5,6,8,12,15]	[1,2,5,6,8,12,15]
[1,2,3,4]	[1,2,3,4]

3. getBMR

- This function is used to calculate the Basal Metabolic Rate.

```
118 ⑥ @  public double getBMR(@RequestBody Map<String, String> body){  
119  
120      double height = Double.parseDouble( body.get("height")); //cms  
121  
122      double weight = Double.parseDouble( body.get("weight")); //kgs  
123  
124      int age = Integer.parseInt(body.get("age")); // years  
125  
126      String gender = body.get("gender"); // male , female  
127  
128      if (height < 0 || weight < 0 || age<0 ) {...}  
133  
134      double bmr=0.0;  
135  
136      if(gender=="male")  
137      {...}  
141      else if(gender=="female")  
142      {...}  
146      else  
147      {...}|  
152  
153      logger.info("[BMR] - INPUT:" +height+"^"+weight+ "^^"+age+"^"+gender+ " ", OUTPUT:" +bmr);  
154  
155      return bmr ;  
156 }
```

CFG and Block Number



Testcase

```
Run: HealthTest.BMR.x
Run: HealthTest (com.example.Soft 1 sec 171 ms)
  ✓ BMR 1 sec 171 ms
    26/Nov/2022:15:35:31 +0530 [HealthController.java] [INFO] HealthController [BMR] - INPUT:180.0^-1.0^18^male, OUTPUT:Invalid
    26/Nov/2022:15:35:31 +0530 [HealthController.java] [INFO] HealthController [BMR] - INPUT:180.0^40.0^18^ma, OUTPUT:Invalid
    26/Nov/2022:15:35:31 +0530 [HealthController.java] [INFO] HealthController [BMR] - INPUT:180.0^55.0^18^male, OUTPUT:1590.0
    26/Nov/2022:15:35:31 +0530 [HealthController.java] [INFO] HealthController [BMR] - INPUT:160.0^60.0^20^female, OUTPUT:1336.0

Process finished with exit code 0
```

Testpath

4 test paths are needed for Edge Coverage

- [1,2,3,4]
- [1,2,5,6,8,10,13]
- [1,2,5,6,7,11,12]
- [1,2,5,6,8,9,11,12]

4 test paths are needed for Prime Path Coverage

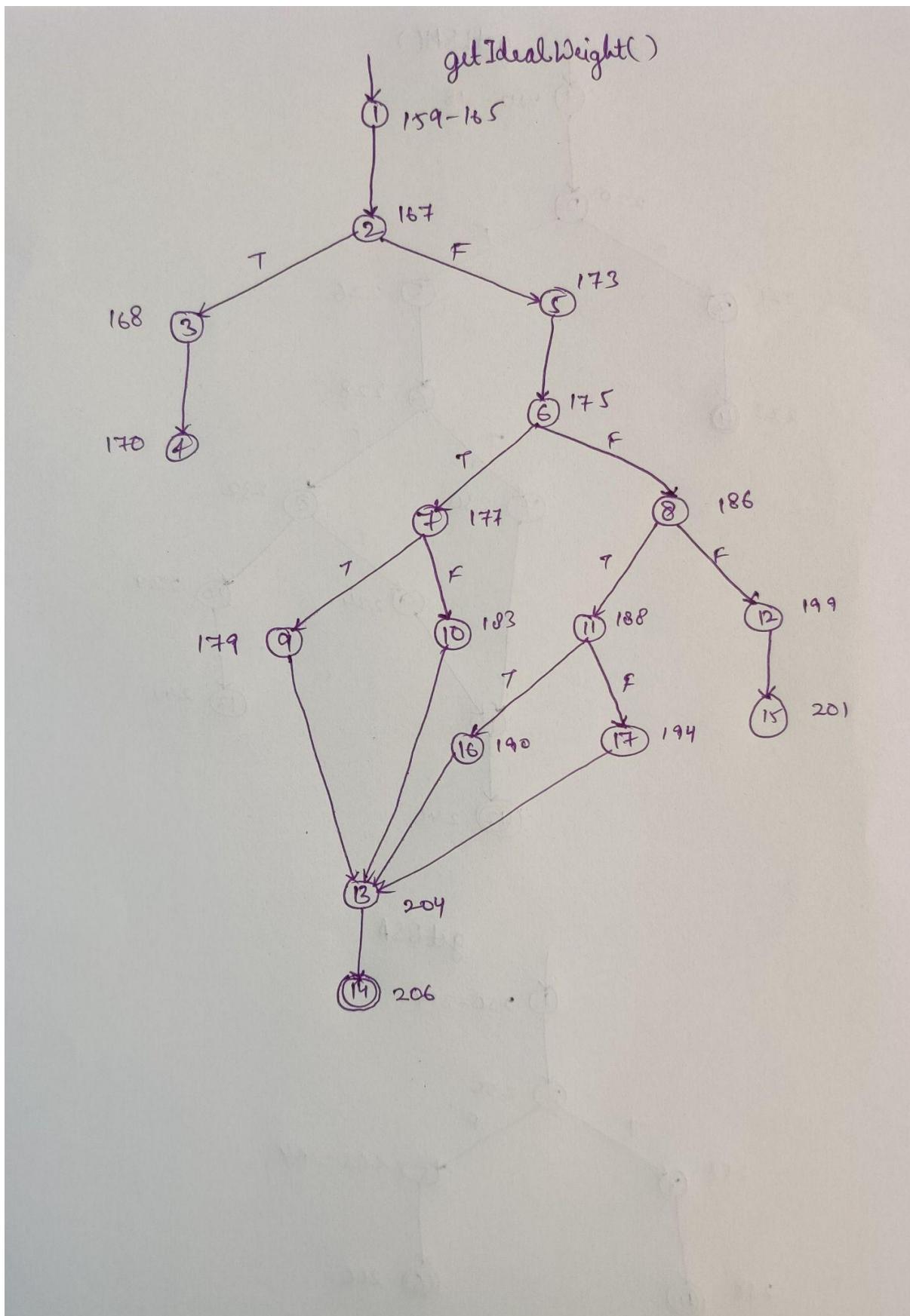
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,8,9,11,12]	[1,2,5,6,8,9,11,12]
[1,2,5,6,7,11,12]	[1,2,5,6,7,11,12]
[1,2,5,6,8,10,13]	[1,2,5,6,8,10,13]
[1,2,3,4]	[1,2,3,4]

4. getIdealWeight

- This function is used to calculate the Basal Metabolic Rate.

```
159  public double getIdealWeight(@RequestBody Map<String, String> body){  
160  
161      double height = Double.parseDouble( body.get("height")); //cms  
162  
163      int age = Integer.parseInt(body.get("age"));// years  
164  
165      String gender = body.get("gender"); // male , female  
166  
167      if (height < 0 || age<0 ) {...}  
172  
173      double weight = 0.0; //kgs  
174  
175      if(gender=="male")  
176      {  
177          if (height<152.4)  
178          {...}  
181          else  
182          {...}  
185      }  
186      else if(gender=="female")  
187      {  
188          if (height<152.4)  
189          {...}  
192          else  
193          {...}  
196      }  
197      else  
198  }
```

CFG and Block Number



Testcase

```
Run: HealthTest.IdealWeight ×
  Tests passed: 1 of 1 test - 1 sec 205 ms
  HealthTest (com.example.Soft 1 sec 205 ms)
    IdealWeight 1 sec 205 ms
      C:\Users\dada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
      26/Nov/2022:15:35:58 +0530 [HealthController.java] [INFO] HealthController [IdealWeight] - INPUT:-180.0^18^male, OUTPUT:Inval
      26/Nov/2022:15:35:58 +0530 [HealthController.java] [INFO] HealthController [IdealWeight] - INPUT:180.0^18^ma, OUTPUT:Inval
      26/Nov/2022:15:35:58 +0530 [HealthController.java] [INFO] HealthController [IdealWeight] - INPUT:140.0^18^male, OUTPUT:34.
      26/Nov/2022:15:35:58 +0530 [HealthController.java] [INFO] HealthController [IdealWeight] - INPUT:180.0^10^male, OUTPUT:78.
      26/Nov/2022:15:35:58 +0530 [HealthController.java] [INFO] HealthController [IdealWeight] - INPUT:140.0^10^female, OUTPUT:7
      26/Nov/2022:15:35:58 +0530 [HealthController.java] [INFO] HealthController [IdealWeight] - INPUT:180.0^10^female, OUTPUT:7

  Process finished with exit code 0
```

Testpath

6 test paths are needed for Edge Coverage

- [1,2,3,4]
- [1,2,5,6,8,12,15]
- [1,2,5,6,7,9,13,14]
- [1,2,5,6,7,10,13,14]
- [1,2,5,6,8,11,17,13,14]
- [1,2,5,6,8,11,16,13,14]

6 test paths are needed for Prime Path Coverage

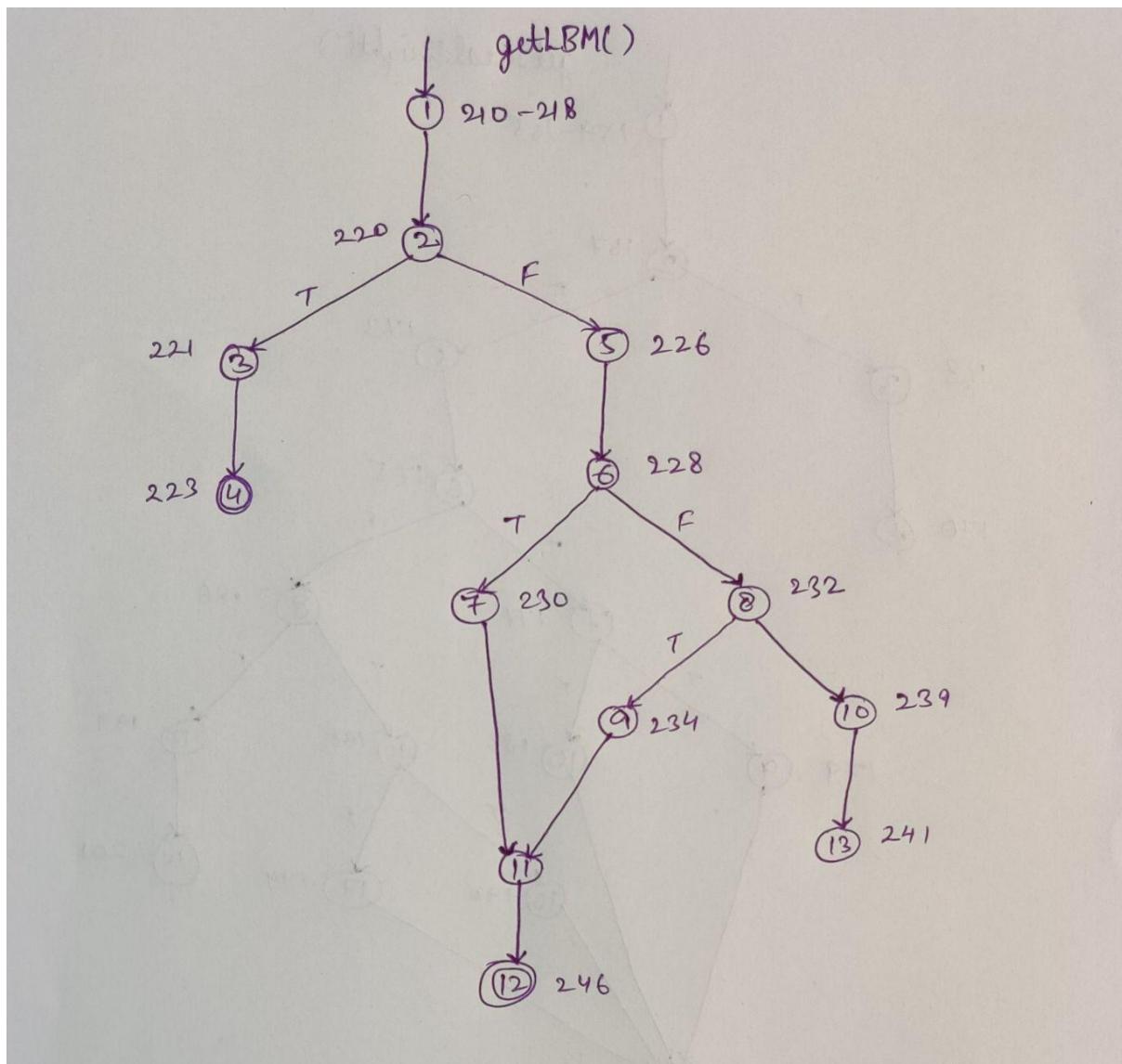
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,8,11,16,13,14]	[1,2,5,6,8,11,16,13,14]
[1,2,5,6,8,11,17,13,14]	[1,2,5,6,8,11,17,13,14]
[1,2,5,6,7,9,13,14]	[1,2,5,6,7,9,13,14]
[1,2,5,6,7,10,13,14]	[1,2,5,6,7,10,13,14]
[1,2,5,6,8,12,15]	[1,2,5,6,8,12,15]
[1,2,3,4]	[1,2,3,4]

5. getLBM

- This function is used to calculate the Lean Body Mass.

```
210 ⑥ @    public double getLBM(@RequestBody Map<String, String> body){  
211  
212      double height = Double.parseDouble( body.get("height")); //cms  
213  
214      double weight = Double.parseDouble( body.get("weight")); //kgs  
215  
216      int age = Integer.parseInt(body.get("age")); // years  
217  
218      String gender = body.get("gender"); // male , female  
219  
220      if (height < 0 || weight < 0 || age<0 ) {...}  
221  
222      double lbum=0.0;  
223  
224      if(gender=="male")  
225      {...}  
226      else if(gender=="female")  
227      {...}  
228      else  
229      {...}  
230  
231      logger.info("[LBM] - INPUT:" +height+"^"+weight+ " ^ "+age+"^"+gender+ " , OUTPUT:" +lbum);  
232  
233      return lbum ;  
234 }  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247
```

CFG and Block Number



Testcase

Screenshot of an IDE showing the results of a test run for `HealthTest.LBM`. The test passed in 1 second.

Output log:

```
C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
26/Nov/2022:15:36:39 +0530 [HealthController.java] [INFO] HealthController [LBM] - INPUT:180.0^1.0^18^male, OUTPUT:Invalid
26/Nov/2022:15:36:39 +0530 [HealthController.java] [INFO] HealthController [LBM] - INPUT:180.0^40.0^18^ma, OUTPUT:Invalid
26/Nov/2022:15:36:39 +0530 [HealthController.java] [INFO] HealthController [LBM] - INPUT:180.0^55.0^18^male, OUTPUT:51.24
26/Nov/2022:15:36:39 +0530 [HealthController.java] [INFO] HealthController [LBM] - INPUT:160.0^60.0^20^female, OUTPUT:42.

Process finished with exit code 0
```

Testpath

4 test paths are needed for Edge Coverage
[1,2,3,4]
[1,2,5,6,8,10,13]
[1,2,5,6,7,11,12]
[1,2,5,6,8,9,11,12]

4 test paths are needed for Prime Path Coverage

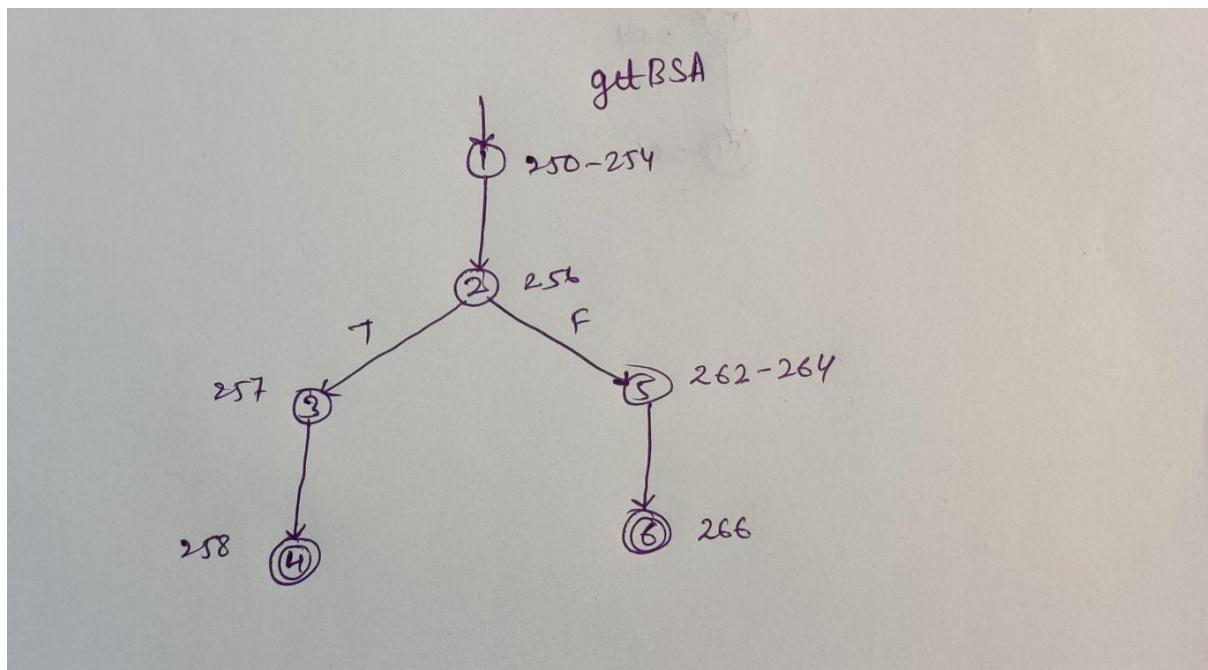
Test Paths	Test Requirements that are toured by test paths directly
[1,2,5,6,8,9,11,12]	[1,2,5,6,8,9,11,12]
[1,2,5,6,7,11,12]	[1,2,5,6,7,11,12]
[1,2,5,6,8,10,13]	[1,2,5,6,8,10,13]
[1,2,3,4]	[1,2,3,4]

6. getBSA

- This function is used to calculate the Body Surface Area.

```
250 ⑥ @ ↴    public double getBSA(@RequestBody Map<String, String> body){  
251  
252      double height = Double.parseDouble( body.get("height")); //cms  
253  
254      double weight = Double.parseDouble( body.get("weight")); //kgs  
255  
256      if (height < 0 || weight < 0 ) {  
257          logger.info("[BSA] - INPUT:" +height+"^"+weight+ " , OUTPUT:" + "Invalid Input");  
258  
259          return -1;  
260      }  
261  
262      double bsa= 0.016667 * Math.sqrt(weight) * Math.sqrt(height);  
263  
264      logger.info("[BSA] - INPUT:" +height+"^"+weight+ " , OUTPUT:" +bsa);  
265  
266      return bsa ;  
267  }
```

CFG and Block Number



Testcase

Screenshot of a Java IDE showing the results of a test run for "HealthTest.BSA".

Run: HealthTest.BSA X

Tests passed: 1 of 1 test – 1 sec 256 ms

HealthTest (com.example.Soft 1 sec 256 ms) C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
└─ BSA 1 sec 256 ms 26/Nov/2022:15:37:05 +0530 [HealthController.java] [INFO] HealthController [BSA] - INPUT:180.0^75.0, OUTPUT:Invalid Input
26/Nov/2022:15:37:05 +0530 [HealthController.java] [INFO] HealthController [BSA] - INPUT:180.0^40.0, OUTPUT:1.414241846644
Process finished with exit code 0

Testpath

2 test paths are needed for Edge Coverage

[1,2,5,6]

[1,2,3,4]

2 test paths are needed for Prime Path Coverage

Test Paths	Test Requirements that are toured by test paths directly
[1,2,3,4]	[1,2,3,4]
[1,2,5,6]	[1,2,5,6]

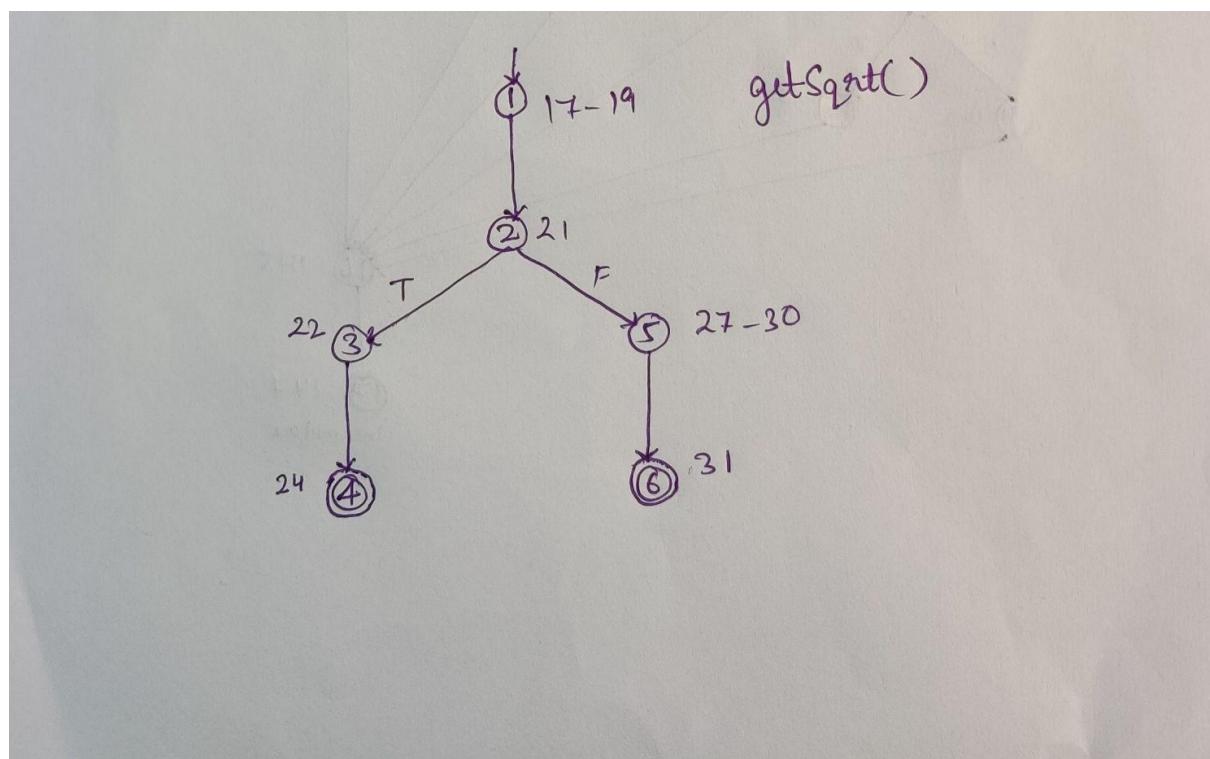
III. Scientific Calculator

1. get.Sqrt

- This function is used to calculate the square root of a number.

```
17 ⑥ @    public double get.Sqrt(@RequestBody Map<String, String> body){  
18  
19      double input1 = Double.parseDouble(body.get("input1"));  
20  
21      if (input1 < 0 ) {  
22          logger.info("[SQRT] - INPUT:" + input1 + " , OUTPUT:" + "Invalid Input");  
23  
24          return -1;  
25      }  
26  
27      double res = Math.sqrt(input1) ;  
28  
29      logger.info("[SQRT] - INPUT:" + input1 + " , OUTPUT:" + res);  
30  
31      return res ;  
32  }
```

CFG and Block Number



Testcase

```
Run: CalTest.squareRoot
Tests passed: 1 of 1 test - 1 sec 102 ms
C:\Users\drada\jdks\openjdk-18.0.1.1\bin\java.exe ...
26/Nov/2022:15:32:06 +0530 [CalController.java] [INFO] CalController [SQRT] - INPUT:-1.0 , OUTPUT:Invalid Input
26/Nov/2022:15:32:06 +0530 [CalController.java] [INFO] CalController [SQRT] - INPUT:25.0 , OUTPUT:5.0
Process finished with exit code 0
```

Testpath

2 test paths are needed for Edge Coverage
[1,2,5,6]
[1,2,3,4]

2 test paths are needed for Prime Path Coverage

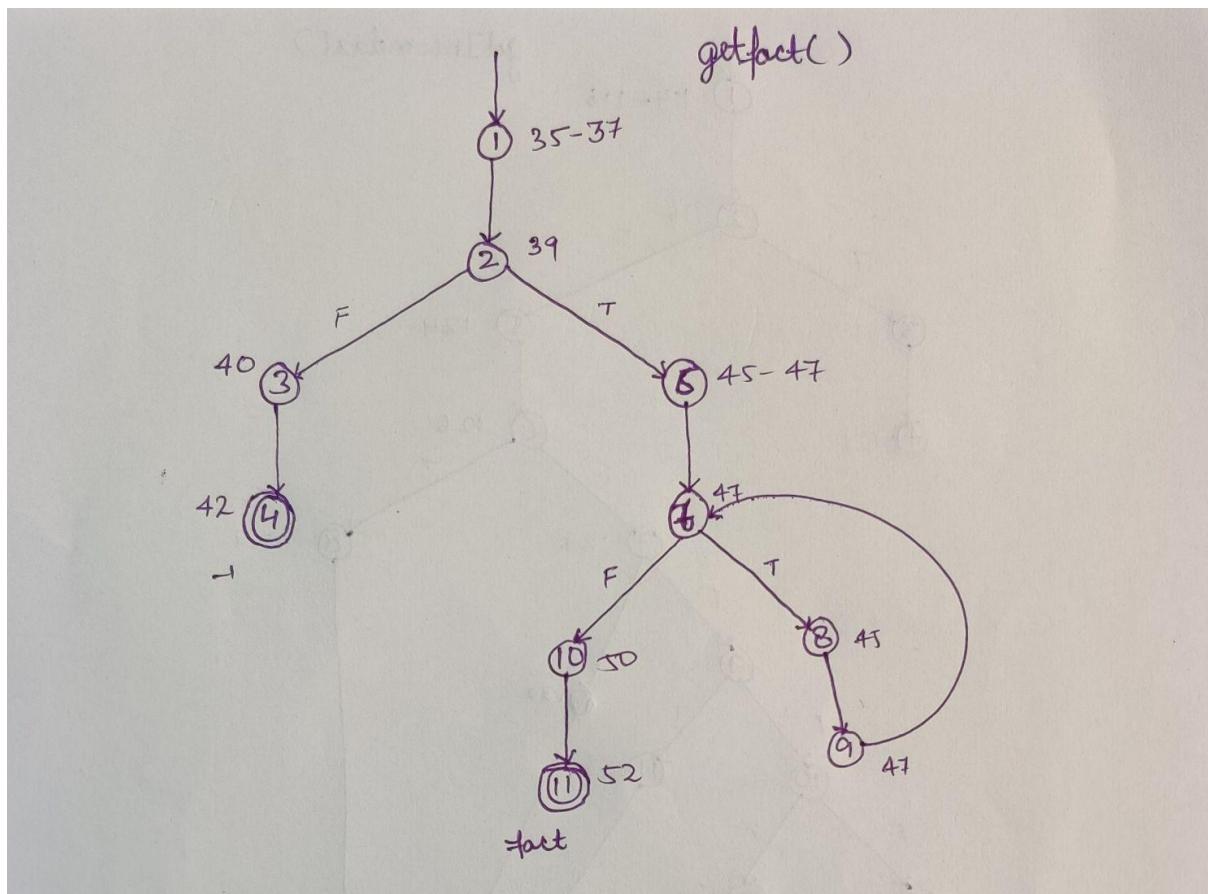
Test Paths	Test Requirements that are toured by test paths directly
[1,2,3,4]	[1,2,3,4]
[1,2,5,6]	[1,2,5,6]

2. getFact

- This function is used to calculate the factorial of a number.

```
35 public double getFact(@RequestBody Map<String, String> body){
36
37     double input1 = Double.parseDouble( body.get("input1"));
38
39     if (input1 < 0 ) {
40         logger.info("[FACT] - INPUT:" + input1 + " , OUTPUT:" + "Invalid Input");
41
42         return -1;
43     }
44
45     double res= 1;
46
47     for(double i=1;i<=input1;i++)
48         res *= i;
49
50     logger.info("[FACT] - INPUT:" + input1 + " , OUTPUT:" + res);
51
52     return res;
53 }
```

CFG and Block Number



Testcase

```

Run: CalTest.factorial x
  ✓ Tests passed: 1 of 1 test – 1 sec 858 ms
  ✓ CalTest (com.example.Softwar 1 sec 858 ms C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ...
    factorial 1 sec 858 ms
      26/Nov/2022:15:33:16 +0530 [CalController.java] [INFO] CalController [FACT] - INPUT:-6.0 , OUTPUT:Invalid Input
      26/Nov/2022:15:33:17 +0530 [CalController.java] [INFO] CalController [FACT] - INPUT:1.0 , OUTPUT:1.0
      26/Nov/2022:15:33:17 +0530 [CalController.java] [INFO] CalController [FACT] - INPUT:2.0 , OUTPUT:2.0
  Process finished with exit code 0
  
```

Testpath

2 test paths are needed for Edge Coverage

[1,2,3,4]

[1,2,6,7,8,9,7,10,11]

3 test paths are needed for Prime Path Coverage

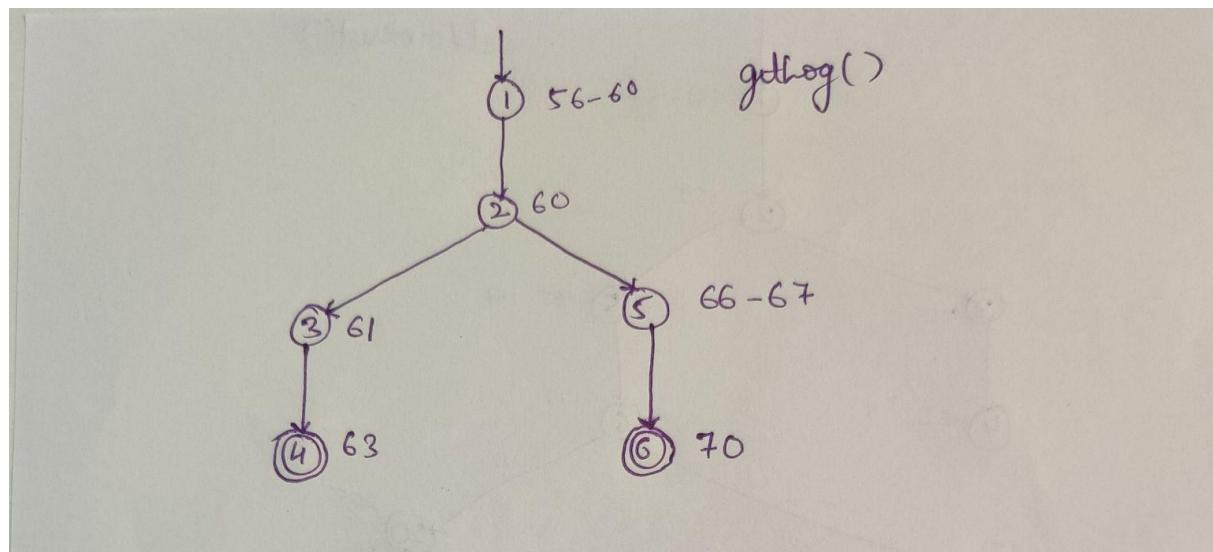
Test Paths	Test Requirements that are toured by test paths directly
[1,2,6,7,10,11]	[1,2,6,7,10,11]
[1,2,3,4]	[1,2,3,4]
[1,2,6,7,8,9,7,8,9,7,10,11]	[1,2,6,7,8,9], [8,9,7,10,11], [9,7,8,9], [8,9,7,8], [7,8,9,7]

3. getLog

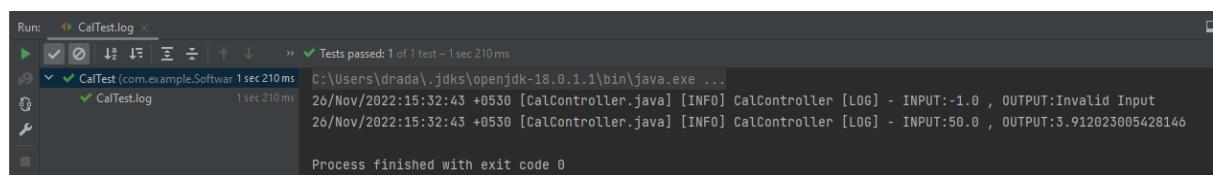
- This function is used to calculate the logarithm value of a number.

```
56  @
57
58      double input1 = Double.parseDouble(body.get("input1"));
59
60      if (input1 < 0 ) {
61          logger.info("[LOG] - INPUT:" + input1 + " , OUTPUT:" + "Invalid Input");
62
63          return -1;
64      }
65
66      double res = Math.log(input1);
67
68      logger.info("[LOG] - INPUT:" + input1 + " , OUTPUT:" + res);
69
70      return res ;
71 }
```

CFG and Block Number



Testcase



Testpath

2 test paths are needed for Edge Coverage
[1,2,5,6]
[1,2,3,4]

2 test paths are needed for Prime Path Coverage

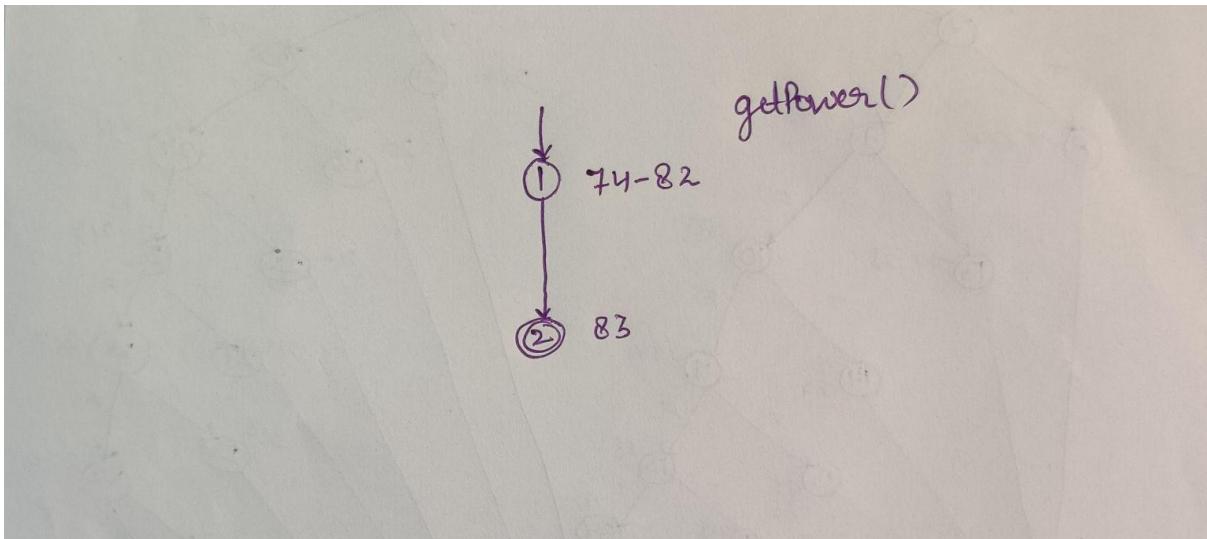
Test Paths	Test Requirements that are toured by test paths directly
[1,2,3,4]	[1,2,3,4]
[1,2,5,6]	[1,2,5,6]

4. getPower

- This function is used to calculate the power of a number.

```
74 ⑥ @ ublic double getPower(@RequestBody Map<String, String> body){  
75  
76      double input1 = Double.parseDouble(body.get("input1"));  
77  
78      double input2 = Double.parseDouble( body.get("input2"));  
79  
80      double res = Math.pow(input1,input2);  
81  
82      logger.info("[POWER] - INPUT:" + input1+"^"+input2+" , OUTPUT:" + res);  
83  
84      return res ;  
85 }
```

CFG and Block Number



Testcase

A screenshot of a terminal window titled "Run: CalTest.power". The window shows the output of a test run. It includes the command used ("java -jar CalTest.jar power"), the execution time ("1 sec 257 ms"), the path taken ("C:\Users\drada\.jdks\openjdk-18.0.1.1\bin\java.exe ..."), the date and time of execution ("26/Nov/2022:15:33:56 +0530 [CalController.java] [INFO] CalController [POWER] - INPUT:10.0^2.0 , OUTPUT:100.0"), and the message "Process finished with exit code 0".

Testpath

1 test path is needed for Edge Coverage
[1,2]

1 test path is needed for Prime Path Coverage

Test Paths	Test Requirements that are toured by test paths directly
[1,2]	[1,2]

Contributions

- Code, CFGs, Test case design, Edge Coverage, Prime Path Coverage of functions present in Bank Calculator Module and Scientific Calculator Module is done by **Divyesh Radadiya (MT2021044)**.
- Code, CFGs, Test case design, Edge Coverage, Prime Path Coverage of functions present in Health Calculator Module and Scientific Calculator Module is done by **Prashant Chaudhary (MT2021102)**.