refined-randoop1

Strategy

1. Run the current test suites and obtain the coverage results
2. Record the methods with low coverage and ask LLMs to generate tests for them
3. Manually check and filter the generated tests
4. Add the checked generated tests to all the test suites

Result Comparison

The refined tests can locate bugs in branches not covered by the original test suites. For example, `reverse` has 0 score in original tests, but is detected as a potential bug in refined tests:
Original:

```
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>
$stack5 = lengthof array        0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       from =
from + 1        0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       goto
[?= staticinvoke <comp5111.assignment.Counter: void invocateStmt(java.lang.String)>("
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>6")]
0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       if 0 >
from goto return null   0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       if
from > to goto return null      0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       if
from >= to#5 goto return array  0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       if to
< $stack5 goto to#5 = to + -1   0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       return
array   0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       return
null    0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       to#5 =
to + -1 0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       to#5 =
to#5 + -1       0.000000        799
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>
virtualinvoke this.<comp5111.assignment.cut.Subject$SortTools: int[]
swap(int[],int,int)>(array, from, to#5) 0.000000        799
```

Refined:

```
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       from =
from + 1        0.176777        308
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       goto
[?= staticinvoke <comp5111.assignment.Counter: void invocateStmt(java.lang.String)>("
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>6")]
0.176777        308
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>       to#5 =
to#5 + -1       0.176777        308
<comp5111.assignment.cut.Subject$SortTools: int[] reverse(int[],int,int)>
virtualinvoke this.<comp5111.assignment.cut.Subject$SortTools: int[]
swap(int[],int,int)>(array, from, to#5) 0.176777        308
```

The refined tests detect three different bugs ignored by original tests, which are located in `reverse`, `lower` and `upper`.