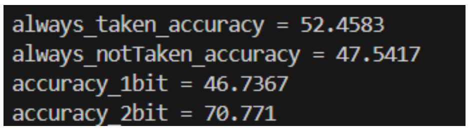
REPORT

BRANCH PREDICTOR

A. Bubble_Test_Lab



```
always_taken_accuracy = 52.4583
always_notTaken_accuracy = 47.5417
accuracy_1bit = 46.7367
accuracy_2bit = 70.771
```

Out of these four, the 2-bit predictor has the highest accuracy (70.771%) followed by the always taken predictor (52.4583%). The 1-bit predictor and the always not taken predictor have the lowest accuracies (around 47%).

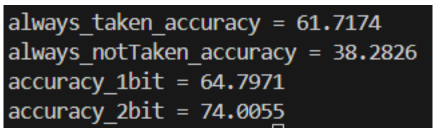
There are a couple of reasons why this might have happened:

Looping behavior: Bubble sort is a sorting algorithm that involves a loop that iterates repeatedly. This repetitive nature means that many of the branches in the assembly trace will be taken repeatedly. A 2-bit predictor, which can better handle these repetitive branches, will outperform the simpler predictors.

Limited history: A 1-bit predictor only looks at the outcome of the most recent branch to make a prediction. In the case of bubble sort, there might be longer range correlations between branches that a 2-bit predictor can exploit to improve its accuracy.

Overall, the results you obtained are consistent with what we would expect from different branch predictors. The 2-bit predictor, which can capture more history and exploit loop behavior, was able to achieve the highest accuracy.

B. SQRT_Test_Lab



```
always_taken_accuracy = 61.7174
always_notTaken_accuracy = 38.2826
accuracy_1bit = 64.7971
accuracy_2bit = 74.0055
```

Out of these four, the 2-bit predictor has the highest accuracy (74.0055%) followed by the 1-bit predictor (64.7971%). The always taken predictor has an accuracy of 61.7174%, and the always not taken predictor has the lowest accuracy (38.2826%).

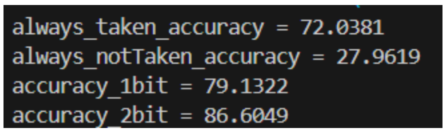
There are a couple of reasons why this might have happened:

Branch behavior: The sqrt trace might have a more balanced distribution of taken and not taken branches. In this case, a 2-bit predictor which can record both taken and not taken would be more accurate than the always taken or always not taken predictors.

Correlation between branches: A 2-bit predictor can also exploit correlations between branches. If the outcome of a branch is correlated with the outcome of the previous branch, the 2-bit predictor can learn this correlation and improve its accuracy.

Overall, the results you obtained are consistent with what we would expect from different branch predictors. The more sophisticated 2-bit predictor was able to achieve the highest accuracy.

C. Recursion_Test_Lab



```
always_taken_accuracy = 72.0381
always_notTaken_accuracy = 27.9619
accuracy_1bit = 79.1322
accuracy_2bit = 86.6049
```

As expected, the more sophisticated branch predictors outperform the simpler ones.

Limited benefit of always taken/not taken: The always taken predictor does well (72.0381%) because most branches in recursion are likely taken (recursive calls). However, it misses the base cases where the recursion ends (not taken branch). The always not taken predictor performs poorly (27.9619%) because it misses all the recursive calls.

1-bit predictor captures local history: The 1-bit predictor (79.1322%) significantly improves over always taken by learning from the previous branch. It can identify patterns like taken branches followed by another taken branch (recursive calls) and not taken branches at the end of recursion.

2-bit predictor captures more complex patterns: The 2-bit predictor (86.6049%) achieves the highest accuracy. It can not only learn from the previous branch but also potentially capture even longer-range patterns in the branching behavior of the recursion. This could include things like the depth of the recursion or specific sequences of taken/not taken branches.

CONCLUSION:

In conclusion, the results demonstrate the effectiveness of using more sophisticated branch predictors for improving prediction accuracy. The 2-bit predictor's ability to capture more history and complex patterns in the branching behavior of the recursive trace leads to its superior performance.