Testing

• Lots of testing methodologies.
  • Unit Testing (Usually associate with coding)
  • System Testing (Usually associated with system integration)
• These are basic testing methods.
• Balanced Expressions Task is a kind of integration task.
• Testing Stack operations is a kind of unit testing.
Test Driven Development (TDD)

• List your expectations
  • These are usually mentioned in your homework.

• When to design test cases
  • While reading your homework?
  • Test cases keep changing as you gain more understanding about the problem.

• When to write test code
  • Either while writing your implementation.
  • Or even before you start implementation.
Unit Testing

• Test each of various flows (conditions) in your code.
• List at least one scenario for each condition, and write test code for the same.
• Example:
  • For Dequeue - insertBackList
  • Inserting into full dequeue.
  • Inserting into non-empty dequeue.
  • Inserting into empty dequeue.
Usual Test Scenarios

• Positive cases
  • Does your sort algorithm sort all input numbers?

• Negative cases
  • Does your balanced expression validate unbalanced expressions?

• Edge cases
  • Sorting empty input array?
  • Inserting into full stack?
  • Testing sort routine on already sorted input.
Usual Test Scenarios

- System state after the calls
  - Does your algorithm work as expected on consecutive calls?
    - On at least 40% of homework solutions two consecutive calls did not work as expected.
  - Is the system state as expected after the call?
    - System state after inserting the same record 100 times?
    - Test it when you implement Bag!
- Has your application freed all the memory?
  - Test it with `valgrind --tool=memcheck stackapp` (for example.)
Testing Overhead

- On a properly automated testing system
  - Test code is usually 5-10 times more than actual code (which needs to be tested).
  - Its truly a lot of overhead on the developer!!
  - Amount of testing needed depends on how critical the system is.....
  - Criticality of a system is usually reflected in the number of test scenarios one comes up with...
  - “Criticality” is subjective unless there is something to lose!!
Simple Test code

Remember the `isBalanced` call?

```c
char *e1 = "{{()[[]]}";
char *e2 = "{{{{";
char *e3 = "}}}}}";
char *e4 = "{{[[[]]}}";
char *e5 = "{{[(())]}";

print_message(isBalanced(e1), e1);
print_message(isBalanced(e2), e2);
print_message(isBalanced(e3), e3);
print_message(isBalanced(e4), e4);
print_message(isBalanced(e5), e5);
```

Are these test cases sufficient?

What about empty string?
Good Luck