NSTC

Refinement paradigm
Description of Refinement Paradigm

- A pattern/method to produce correct code that can simplify existing program while maintaining their formal verification.
- We need ensure that we can get the correct result for all Inputs
- Safety-critical systems/business-critical systems where it's very important to getting things ALWAYS right.
- The consequences of error can be very severe! (DEATH!!!
Exploring how RP works

- Start with your problem formulated in some kind of logical manner
- Apply refinement techniques one by one to transform the logical representation into code.
- By induction, we can show that the transformation from the logical representation to code does not lose any meaning or exactness.
- Therefore, if the logical representation is correct, then the generated code must also be correct.
What it means to be correct (RP)

- In some contexts, there is a ‘yes’ or ‘no’ answer. This implies that every solution that an algorithm might generate is strictly ‘correct’ or ‘incorrect’.
- Use of the refinement paradigm ensures that your code will only generate solutions that are correct.
What it means to be Correct (NSTC)

- In other contexts, the ‘correctness’ of a solution is on a continuous scale (perhaps from 0% to 100%).
- Non-standard algorithms can be used to give an answer as close to 100% as possible without incurring high cost.
- Sometimes probabilistic, approximate and fuzzy solutions can be just as useful and more efficient.
Why refinement is not always appropriate

- Costly in many ways (Time, effort, resources, skills, etc.) and sometimes costs outweigh the benefits.
- Refinement is important for safety-critical applications, and when correctness is of extreme importance.
- NSTC algorithms may be fine in other cases, especially when only a ‘good enough’ solution is needed and perfection isn’t a requirement.
Problems suitable for RP-generated code

- SAT problem
- Mathematical problems
- Graph Transformations
Problems that may be more suited for NSTC-based solutions

- Vehicle routing algorithms:
  - For getting Grannie to the supermarket, you might not rely on the solution being particularly good (low ‘correctness’ threshold).
  - For getting her home again when she needs the loo, the correctness of the solution might be more important (higher ‘correctness’ threshold).
  - For an ambulance, an even higher threshold might be appropriate.
  - The choice of threshold may come down to a cost-benefit analysis.
- Optimization algorithms
Questions???