## **Bab 1**

### **Latar Belakang**

Sudah menjadi rahasia umum bahwa seorang pemrogram menginginkan programnya bebas dari kesalahan. Banyak waktu yang sudah terhabiskan oleh berbagai perusahaan teknologi untuk memastikan bahwa produk yang mereka rilis tidak memiliki kecacatan berarti yang mungkin saja bisa menelan korban jiwa. Kesalahan atau *bug* yang bisa ditangkap diawal proses pengembangan sebuah program bisa diatasi dengan lebih mudah daripada *bug* yang hanya disadari jauh dalam sebuah proses pengembangan. Salah satu metode yang dikembangkan oleh para ilmuwan komputasi untuk bisa menangkap sebanyak-banyaknya atau bahkan seluruh kesalahan yang mungkin terjadi dalam sebuah program pada tahap yang seawal mungkin adalah Metode Formal.

Metode Formal merupakan sebuah teknik untuk memanfaatkan sistem pembuktian matematika untuk membuktikan bahwa sebuah program akan berjalan sebagimana spesifikasi yang sudah ditetapkan. Teknik ini biasanya dilakukan dengan merepresentasikan bagian-bagian dari spesifikasi kebutuhan yang diinginkan dan program yang dikembangkan ke dalam simbol-simbol matematika yang kemudian akan dibuktikan kecocokannya dengan menggunakan berbagai teknik matematika yang sudah paten. Teknik ini bisa dilakukan baik secara manual maupun secara otomatis dengan menggunakan beberapa program yang sudah dirilis sebelumnya seperti Z3, CVC4, atau MathSat. Program otomatis ini bisa membantu mempercepat proses pencarian *bug* yang sebelumnya berpotensi menjadi sebuah proses *whack-a-mole* tiada akhir menjadi sebuah proses yang bisa dengan mudah mengurangi jumlah kesalahan yang mungkin terjadi dalam program jauh sebelum kesalahan itu biasanya akan disadari dengan metode lain. Faktanya, Metode Formal sudah digunakan dan direkomendasikan oleh baik IEC, ESA, FAA, maupun NASA sebagai sebuah subjek yang harus dikuasai oleh setiap ilmuwan maupun teknisi komputasi berikut dengan kompetensi matematika yang dibutuhan untuk memahaminya (Baier, 2008).

Namun dibalik kelebihan dari Metode Formal juga bersembunyi kekurangannya. Metode Formal bukanlah merupakan sebuah metode yang mudah untuk dipelajari oleh seorang pemrogram yang tidak terlalu tertarik melakukan studi matematika apalagi praktisi pemrograman amatir yang semakin banyak bermunculan di masa ini. Dibutuhkan usaha tambahan yang tidak sepele untuk menganalisis, mengidentifikasi, dan mengubah karakakteristik utama spesifikasi kebutuhan dan program menjadi representasi matematika yang bisa digunakan untuk analisis Metode Formal. Usaha yang dibutuhkan bisa menjadi sangat besar sehingga keuntungan yang bisa diraih dengan menggunakan metode ini dianggap tidaklah lebih baik daripada biaya yang harus dibayar sehingga metode ini hanya digunakan untuk sistem kritikal yang memiliki potensi kerugian yang sangat besar untuk setiap kesalahan yang terjadi sehingga beban untuk melakukan Metode Formal bisa dijustifikasi (Pena, 2016). Terlihatlah bahwa jika seseorang menginginkan metode ini untuk lebih banyak digunakan oleh kalangan pemrogram, maka dibutuhkan cara untuk mengurangi usaha dan beban yang dibutuhkan untuk menggunakan metode ini.

Salah satu metode untuk memudahkan pengembangan metode formal untuk perangkat lunak adalah dengan memungkinkan logika-logika untuk program dituliskan langsung berdampingan dengan program itu sendiri. Salah satu perangkat lunak yang dikembangkan untuk tujuan tersebut adalah Liquid Haskell untuk bahasa pemrograman Haskell. Pada dasarnya bahasa pemrograman berparadigma fungsional sudah mengalami keunggulan untuk pengaplikasian Metode Formal karena sifat bahasa tersebut yang transparan dan tidak menghasilkan “efek samping” saat mengesekusi suatu fungsi sehingga seorang analis bisa dengan mudah melihat efek dari interaksi antara berbagai fungsi tanpa harus mempertimbangkan bahwa hasil dari interaksi tersebut mungkin berbeda dengan input yang sama; sama seperti sebuah fungsi matematika yang konvensional. Karena itu Haskell sebagai sebuah bahasa fungsional murni juga memungkinkan kemudahan pengaplikasian Metode Formal pada program yang dituliskan pada bahasa tersebut. Liquid Haskell memanfaatkan kemudahan ini dengan mengintegrasikan *SMT Solver* seperti Z3 atau CVC4 dan memanfaatkan perangkat tersebut untuk memastikan keabsahan program yang dibuat pada Haskell secara otomatis. Hal ini memungkinkan penulisan spesifikasi langsung pada program (atau dalam file yng berdampingan) dan kemudian melakukan pengecekan otomatis program terhadap spesifikasi yang sudah tertulis sehingga proses Metode Formal menjadi jauh lebih mudah.

Sebagai seorang pelajar dalam ilmu komputasi, mahasiswa S1 Teknik Informatika ITB (IF ITB) juga merupakah sebuah pihak yang akan sangat diuntungkan dalam mempelajari Metode Formal. Namun seperti pemrogram lain yang sudah disebutkan sebelumnya, hal ini merupakan suatu hal yang tidak mudah dilakukan dan waktu yang akan dihabiskan mungkin saja lebih baik digunakan untuk mengajarkan materi studi lain. Menggunakan kemudahan yang ditawarkan oleh Liquid Haskell yang sudah disebutkan sebelumnya, diharapkan mahasiswa IF ITB dapat memahami materi Metode Formal ini dalam waktu yang lebih mudah untuk dialokasikan serta dapat memberikan pemahaman yang lebih mumpuni.

### **Rumusan Masalah**

Berdasarkan latar belakang dan fokus masalah tersebut, terumuskan beberapa masalah yang sebaiknya diselesaikan:

1. Bagaimanakah sistem terbaik untuk menjelaskan Metode Formal kepada Mahasiswa S1 Teknik Informatika ITB dengan menggunakan Liquid Haskell?
2. Bagaimanakah penyusunan sistem materi yang baik untuk diajarkan dalam sistem ini?
3. Apakah sistem yang dikembangkan benar-benar mampu mempermudah pengguna untuk memahami Metode Formal?

### **Tujuan**

Penelitian ini bertujuan untuk:

1. Melakukan analisis materi Metode Formal yang bisa disarikan sebagai materi dalam *learning tools*.
2. Melakukan implementasi *learning tools* Metode Haskell dengan berbasis Haskell dan Liquid Haskell.

### **Batasan Masalah**

Batasan permasalahan dalam penelitian ini adalah sebagai berikut:

1. Target subjek pengajaran ini adalah Mahasiswa S1 Teknik Informatika ITB ataupun seorang pemrogram yang memiliki kompetensi dan pengalaman dalam membuat program dan mengetahui bahasa pemrograman Haskell.
2. Materi tidak ditujukan untuk mengajarkan teknik pembuktian matematika Metode Formal secara mendetil namun lebih merupakan sebuah panduan sederhana dalam menggunakan Metode Formal untuk meningkatan akurasi program yang ditujukan untuk pemrogram yang tidak membutuhkan pengetahuan matematika mendalam untuk memahami materi tersebut.

### **Metodologi**

Metodologi pengerjaan penelitian ini adalah sebagai berikut:

1. Studi Literatur

Pada tahap ini dilakukan penelitian mendalam terhadap Metode Formal dan Liquid Haskell dengan menggunakan berbagai referensi baik daring maupun luring serta melakukan konsultasi kepada ahli dalam bidang yang berkaitan.

1. Perancangan sistem

Dilakukan analisis serta seleksi materi Metode Formal yang cocok untuk diajarkan serta ditentukan *tools* terbaik yang akan digunakan sebagai dasar untuk pembuatan sistem.

1. Implemetasi sistem

Dilakukan implementasi sistem dengan menggunakan *tools* yang sudah ditentukan dan menggunakan materi yang sudah terseleksi.

### **Jadwal Pelaksaan Tugas Akhir**