INFOSOFT IT SOLUTIONS

Training | Projects | Placements

Revathi Apartments, Ameerpet, 1st Floor, Opposite Annapurna Block,

Info soft It solutions, Software Training & Development 905968394,918254087

HASKELL PROGRAMING TRAINING

1: Introduction to Haskell

- **Overview of Functional Programming**
  - Comparison with imperative programming
  - Benefits of functional programming

2 Basic Syntax and Structure

  - Haskell syntax
  - Haskell's type system
  - Basic data types (Int, Char, Bool, etc.)

3: Functions and Pattern Matching

- **Defining Functions**
  - Function syntax
  - Function application
- **Pattern Matching**
  - Basics of pattern matching
  - Using pattern matching in function definitions
  - Guards and where clauses

4: Lists and Tuples

- **List Operations**
  - Constructing lists
  - Common list functions (head, tail, length, map, filter)
- **List Comprehensions**
  - Syntax and usage
- **Tuples**
  - Definition and usag

- Tuple operations

5: Higher-Order Functions

- **Understanding Higher-Order Functions**
  - Functions as first-class citizens
  - Common higher-order functions (map, filter, foldl, foldr)
- **Function Composition**
  - Composing functions using (.)

6: Type Classes

- **Introduction to Type Classes**
  - Defining and using type classes
  - Common type classes (Eq, Ord, Show, Read)
- **Creating Custom Type Classes**

7: Algebraic Data Types (ADTs)

- **Defining ADTs**
  - Sum types (Either, Maybe)
  - Product types (tuples, custom data types)
- **Recursive Data Types**
  - Lists and Trees

8: Input/Output (IO)

- **Understanding IO in Haskell**
  - The IO type
  - Basic IO operations (print, getLine)
- **Working with Files**
  - Reading from and writing to files

9: Monads

- **Introduction to Monads**
  - Understanding monads through Maybe and List
  - The Monad type class
- **The IO Monad**
  - Using the IO Monad for side effects
- **Monad Transformers (optional**

10: Functors and Applicative Functors

- **Understanding Functors**
  - The Functor type class
  - Mapping over Functors
- **Applicative Functors**
  - The Applicative type class
  - Using <*> and pure

11: Advanced Types

- **Type Families and GADTs**
  - Understanding and using Type Families
  - Generalized Algebraic Data Types (GADTs)
- **Data Kinds and Type-Level Programming (optional)**

12: Concurrency and Parallelism

- **Concurrency in Haskell**
  - Basic concurrency primitives (forkIO, MVar)
- **Parallelism**
  - Using strategies for parallel computation
  - The Par Monad

13: Error Handling and Testing

- **Error Handling**
  - Using Either and Maybe for error handling
  - The Exception type and Control.Exception module
- **Testing in Haskell**
  - Unit testing with HUnit
  - Property-based testing with QuickCheck

14: Libraries and Frameworks

- **Introduction to Haskell Libraries**
  - Exploring common libraries (text, bytestring, containers)
- **Web Development**
  - Introduction to web frameworks (Yesod, Scotty)
- **Interfacing with Databases**
  - Using database libraries (persistent, sqlite-simple)

15: Performance Optimization

- **Profiling Haskell Programs**
  - Tools and techniques for profiling
- **Optimizing Performance**
  - Understanding space and time complexity
  - Common optimization techniques

1: Advanced Type System Features

- **Generalized Algebraic Data Types (GADTs)**
  - Introduction and syntax
  - Use cases and examples
- **Type Families**
  - Associated types
  - Closed and open type families
- **Data Kinds**
  - Promoting data types to kinds
  - Defining and using kind-polymorphic functions

2: Type-Level Programming

- **Singletons and Dependent Types**
  - Singleton types and promoting values to types
  - Using the `singletons` library
- **Type-Level Computation**
  - Type-level arithmetic
  - Type-level lists and operations

3: Advanced Functional Patterns

- **Lens Library**
  - Understanding lenses, prisms, and traversals
  - Practical examples and use cases
- **Zippers**
  - Concept of zippers and their applications
  - Implementing zippers for different data structures

4: Advanced Monads and Monad Transformers

- **Monad Transformers**

    - Understanding and using common transformers (StateT, ReaderT, ExceptT)
    - Stacking transformers and managing complexity
- **Free Monads**
    - Introduction to free monads
    - Building interpreters and DSLs

5: Concurrency and Parallelism

- **Advanced Concurrency Techniques**
    - Software Transactional Memory (STM)
    - Asynchronous programming with `async` library
- **Parallelism**
    - Strategies for parallel computation
    - Using the `parallel` and `monad-par` libraries

6: Profiling and Optimization

- **Profiling Haskell Programs**
    - Profiling tools (GHC Profiler, ThreadScope)
    - Analyzing and interpreting profiling results
- **Optimization Techniques**
    - Space and time complexity analysis
    - Optimizing with strictness and laziness
    - Understanding and avoiding common performance pitfalls

7: Advanced IO and Interfacing with Other Languages

- **Advanced IO Techniques**
    - Working with handles, buffering, and binary data
    - Network programming with `network` library
- **Foreign Function Interface (FFI)**
    - Calling C functions from Haskell
    - Writing Haskell functions callable from C

8: Advanced Libraries and Frameworks

- **Exploring Advanced Libraries**
    - `aeson` for JSON parsing and encoding
    - `conduit` and `pipes` for streaming data processing
- **Web Development**

    - Deep dive into Yesod or Servant for building web applications
    - Authentication, session management, and RESTful APIs

9: <u>Metaprogramming</u>

- **Template Haskell**
    - Introduction to Template Haskell
    - Writing and using splices and quasiquotes
- **Generic Programming**
    - Understanding and using the `Generics` module
    - Deriving generic instances

10: <u>Error Handling and Advanced Testing</u>

- **Error Handling Patterns**
    - Advanced use of `ExceptT`, `Either`, and `Maybe`
    - Custom error types and structured error handling
- **Advanced Testing Techniques**
    - Property-based testing with QuickCheck
    - Model-based testing and state machine testing
    - Benchmarking with Criterion

11: <u>Domain-Specific Languages (DSLs)</u>

- **Designing DSLs**
    - Embedded vs. external DSLs
    - Building a simple DSL in Haskell
- **Interpreting and Compiling DSLs**
    - Writing interpreters
    - Techniques for compiling DSLs to other representations

12: <u>Advanced Compiler Techniques</u>

- **Understanding GHC Internals**
    - GHC architecture and compilation process
    - Writing GHC plugins
- **Core Language and Optimizations**
    - Working with GHC Core
    - Implementing custom optimizations

13: Advanced Topics in Category Theory

- **Category Theory for Haskell Programmers**
  - Monoidal categories, functors, and natural transformations
  - Adjunctions and monads in category theory
- **Applied Category Theory**
  - Using categories to model and solve problems