1.

Record Nr.	UNISA996213397003316
Titolo	2011 IEEE Biomedical Circuits and Systems Conference
Pubbl/distr/stampa	[Place of publication not identified], : IEEE, 2011
ISBN	1-4577-1470-1
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Bibliographic Level Mode of Issuance: Monograph

2.

Record Nr.	UNINA9910973104603321
Autore	Astborg Johan
Titolo	F# for quantitative finance / / Johan Astborg
Pubbl/distr/stampa	Birmingham : , : Packt Publishing, , 2013
ISBN	9781782164630 1782164634
Edizione	[1st edition]
Descrizione fisica	1 online resource (287 p.)
Collana	Community experience distilled
Disciplina	005.13
Soggetti	FA<U+00cc> (Computer program language) Functional programming languages Programming languages (Electronic computers)
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	Cover; Copyright; Credits; About the Author; About the Reviewers; www.PacktPub.com; Table of Contents; Preface; Chapter 1: Introducing F# using Visual Studio; Introduction; Getting started with Visual Studio; Creating a new F# project; Creating a new project in Visual Studio; Understanding the program template; Adding an F# script file; Understanding F# Interactive; Language overview; Explaining mutability and immutability; Primitive types; Explaining type inference; Explaining functions; Learning about anonymous functions; Explaining higher

order functions; Currying; Investigating lists
Concatenating listsTuples; The pipe operator; Documenting your code; Your first application; The whole program; Understanding the program; Extending the example program; The entire program; The power of prototyping; Functional languages in quantitative finance; Understanding the imperative code and interoperability; Summary; Chapter 2: Learning More About F#; Structuring your F# program; Looking into modules; Using functions and values in modules; Namespaces; Looking deeper inside data structures; Record types; Discriminated unions; Enumerations; Arrays
Interesting functions in an array moduleLists; Pattern matching and lists; Interesting functions in a list module; Sequences; Interesting functions in the sequence module; Sets; Maps; Interesting functions in the map module; Options; Strings; Interesting functions in the string module; Choosing data structures; Arrays; Lists; Sets; Maps; More on functional programming; Recursive functions; Tail recursion; Pattern matching; Incomplete pattern matching; Using guards; Pattern matching in assignment and input parameters; Active patterns; Introducing generics; Lazy evaluation; Units of measure
Asynchronous and parallel programmingEvents; Background workers; Threads; Thread pools; Asynchronous programming; The F# asynchronous workflows; Asynchronous binding; Example of using an async workflow; Parallel programming using TPL; MailboxProcessor; A brief look at imperative programming; Object-oriented programming; Classes; Objects and members; Methods and properties; Overloaded operators; Using XML documentation; Useful XML tags; Typical XML documentation; Summary; Chapter 3: Financial Mathematics and Numerical Analysis; Understanding number representation; Integers; Two's complement
Floating-point numbersThe IEEE 754 floating-point standard; Learning about numerical types in F#; Arithmetic operators; Learning about arithmetic comparisons; Math operators; Conversion functions; Introducing statistics; Aggregate statistics; Calculating the sum of a sequence; Calculating the average of a sequence; Calculating the minimum of a sequence; Calculating the maximum of a sequence; Calculating the variance and standard deviation of a sequence; Looking at an example application; Using the Math.NET library; Installing the Math.NET library; Introduction to random number generation
Pseudo-random numbers

Sommario/riassunto

To develop your confidence in F#, this tutorial will first introduce you to simpler tasks such as curve fitting. You will then advance to more complex tasks such as implementing algorithms for trading semi-automation in a practical scenario-based format.If you are a data analyst or a practitioner in quantitative finance, economics, or mathematics and wish to learn how to use F# as a functional programming language, this book is for you. You should have a basic conceptual understanding of financial concepts and models. Elementary knowledge of the .NET framework would also be helpful.