More On Go / Fiance

1st. the news

1. Google is starting a blockchain/cryptocurrency system.
2. Atleast 30 major elections around the world have been run on the Blockchain. This includes a number of provinces in Spain.

2nd. Purpose of a business

1. To make a profit for the owners of the business.

What is “Fiduciary Responsibility”. It means that you have been placed / are in a position of legal responsibility for managing somebody else’s money.

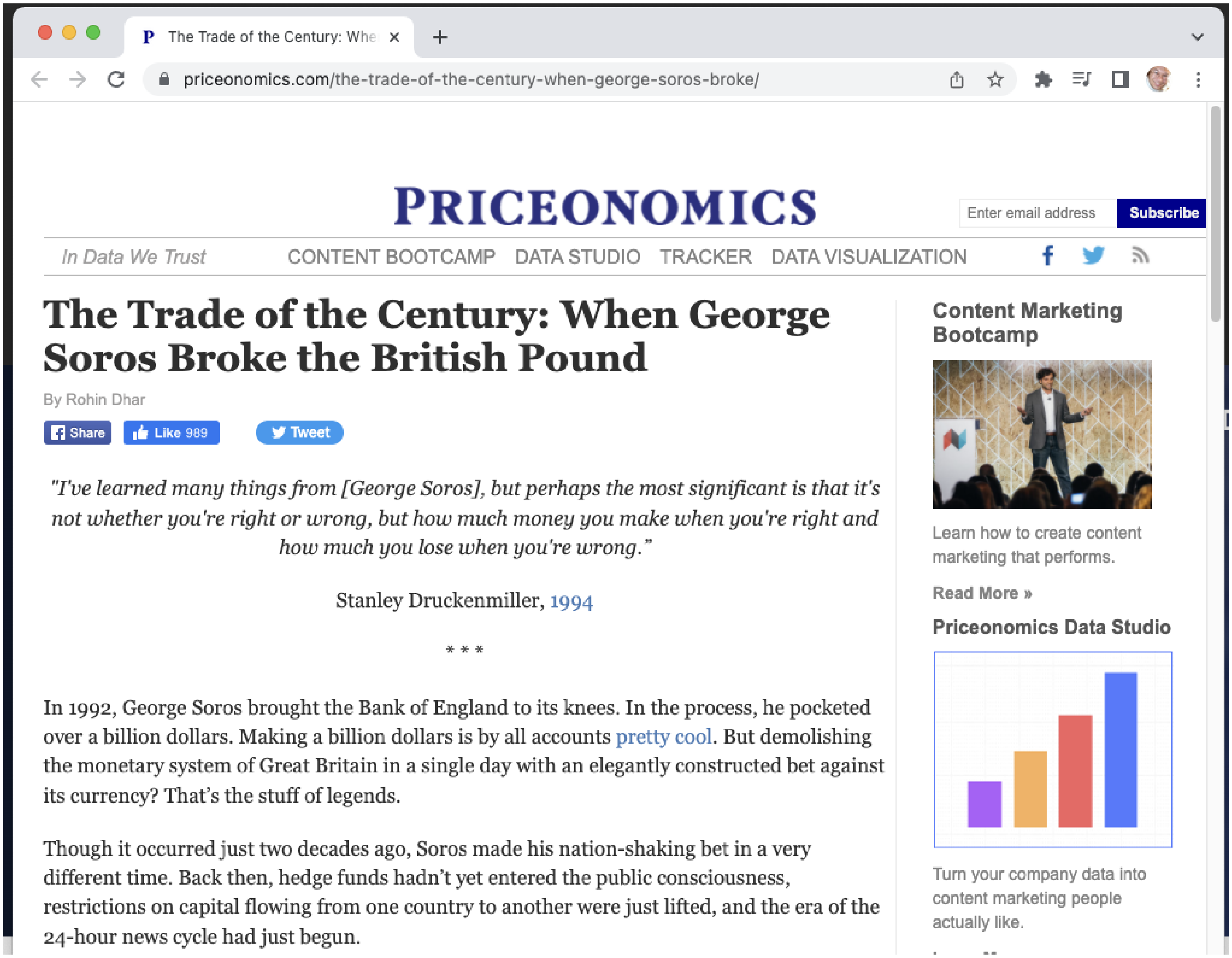
1. Definitions / Economics

“arbitrage” ::= the simultaneous buying and selling of securities, currency, or commodities in different markets or in derivative forms in order to take advantage of differing prices for the same asset.

1. How Soros broke the Bank of England.

Price Stable Cryptocurrences (or pegged cryptocurrencies)

Reference



The Trade of the Century: When George Soros Broke the British Pound

By Rohin Dhar

"I've learned many things from [George Soros], but perhaps the most significant is that it's not whether you're right or wrong, but how much money you make when you're right and how much you lose when you're wrong."

Stanley Druckenmiller, 1994

* * *

In 1992, George Soros brought the Bank of England to its knees. In the process, he pocketed over a billion dollars. Making a billion dollars is by all accounts pretty cool. But demolishing the monetary system of Great Britain in a single day with an elegantly constructed bet against its currency? That's the stuff of legends.

Though it occurred just two decades ago, Soros made his nation-shaking bet in a very different time. Back then, hedge funds hadn't yet entered the public consciousness, restrictions on capital flowing from one country to another were just lifted, and the era of the 24-hour news cycle had just begun.

Content Marketing Bootcamp

Learn how to create content marketing that performs.

Read More »

Priceonomics Data Studio

Turn your company data into content marketing people actually like.

Soros Broke the Bank of England

Personal Finance

Two Economies in US today. The “high-tech” and the “traditional” economy. Per-Capita Income broken down by counties won in 2020.

```
      winner       | per_capita_income_2020
-------------------+------------------------
 Joseph R Biden Jr |               65531.65
 Donald J Trump    |               47960.04
```

GDP Growth broken down by counties won in 2020.

```
      winner       | pct_of_growth
-------------------+---------------
 Donald J Trump    |         9.05%
 Joseph R Biden Jr |        90.95%
```

Blockchain and Mining

What is Mining and How is it implemented.

What is proof-of-work? What is proof-of-stake? What is a public blockchian / private blockchain?

Diagram of blocks

Diagram of mining

What is the mining process

```
package mine

import "github.com/Univ-Wyo-Education/S20-4010/Assignments/02/block"

/*
 import (
     "encoding/hex"
     "fmt"
     "strings"

     block "github.com/Univ-Wyo-Education/S22-4010/homework/02/block"
     "github.com/Univ-Wyo-Education/S22-4010/homework/02/hash"
 )
*/

// MineBlock implements a proof of work mining system where the first 4 digits (2 bytes) of the hex value are
// 0000.  Difficulty can be increaesed by requiring more digits to be 0 or by requring some other pattern to
// apear in the resulting hash.
func InstructorImplementationMineBlock(bk *block.BlockType, difficulty string) {
    // Pseudo-Code
    //
    // 1. Use an infinite loop to:
    //    1. Searilize the data from the block for hasing, Call block.SerializeForSeal to do this.
    //    2. Calculate the hash of the data, Call hash.HashOf to do this. This is the slow part.  What would
    //      happen if we replaced the software with a hash calculator on a graphics card where you could run
    //      4096 hahes at once?  What would happen if we replaced the graphics card with an ASIC - so you had
    //      dedicated hardware to do the hash and you could run 4 billion hashes a second?
    //    3. Convert the hash (it is []byte) to a hex string.  Use the hex.EncodeToString standard go library
    //      function.
    //    4. `fmt.Printf("((Mining)) Hash for Block [%s] nonce [%8d]\r", theHashAsAString, bk.Nonce)`
    //        `\r` will overwrite the same line instead of advancing to the next.
    //    5. See if the first 4 characters of the hash are 0's. - if so we have met the work criteria.
    //        In go this is `if theHashAsAString[0:4] == "0000" {`.  This is create a slice, 4 long from
    //        character 0 with length of 4, then compare that to the string `"0000"`.
    //        - Set the blcoks "Seal" to the hash
    //        - `fmt.Printf("((Mining)) Hash for Block [%s] nonce [%8d]\n", theHashAsAString, bk.Nonce)`
    //          `\n` will overwrite the same *and then advance* to the next line.
    //        - return
    //    5. Increment the Nonce in the block, and...
    //    6. Back to the top of the loop for another try at finding a seal for this block.
    //
    // For the genesis block, when I do this it requires 54586 trips through the loop to calculate the
    // proof of work.
    //
    for {
        // YOur Code at this point ---------------------------------------------------------------------------------
        // seralize the block
        // hash the block
        // convert hash to string. (use hex library package)
        // If the first X characters of the string - match with our passed difficulty (use strings.HasPrefix?)
        //         Set "Seal" field to the hash
        //         Return
        // Increment block Nonce
    }
}
```

1. More on Go

Maps do not synchronize automatically. So... Synchronization Primitives:

```go
package main

import (
    "fmt"
    "sync"
    "time"
)

// SafeCounter is safe to use concurrently.
type SafeCounter struct {
    v   map[string]int
    mux sync.Mutex
}

// Inc increments the counter for the given key.
func (c *SafeCounter) Inc(key string) {
    c.mux.Lock()
    // Lock so only one goroutine at a time can access the map c.v.
    c.v[key]++
    c.mux.Unlock()
}

// Value returns the current value of the counter for the given key.
func (c *SafeCounter) Value(key string) int {
    c.mux.Lock()
    // Lock so only one goroutine at a time can access the map c.v.
    defer c.mux.Unlock()
    return c.v[key]
}

func main() {
    c := SafeCounter{v: make(map[string]int)}
    for i := 0; i < 1000; i++ {
        go c.Inc("somekey")
    }

    time.Sleep(time.Second)
    fmt.Println(c.Value("somekey"))
}
```

A Go Core/Panic

First the Code

```go
package main

import "fmt"

var mm map[string]int

func main() {
    fmt.Println("vim-go")
    mm["bob"] = 3
}
```

Then the bad output.

```
vim-go
panic: assignment to entry in nil map

goroutine 1 [running]:
main.main()
        /Users/philip/go/src/github.com/Univ-Wyo-Education/S22-4010/class/lect/04/go_panic.go:9 +0x71
exit status 2
```