

# **FIN311:人工智能及金融应用**

## **(Artificial Intelligence and Financial Application)**

### **Lecture 1: Introduction of the course**

Xinjie Wang



# Road Map

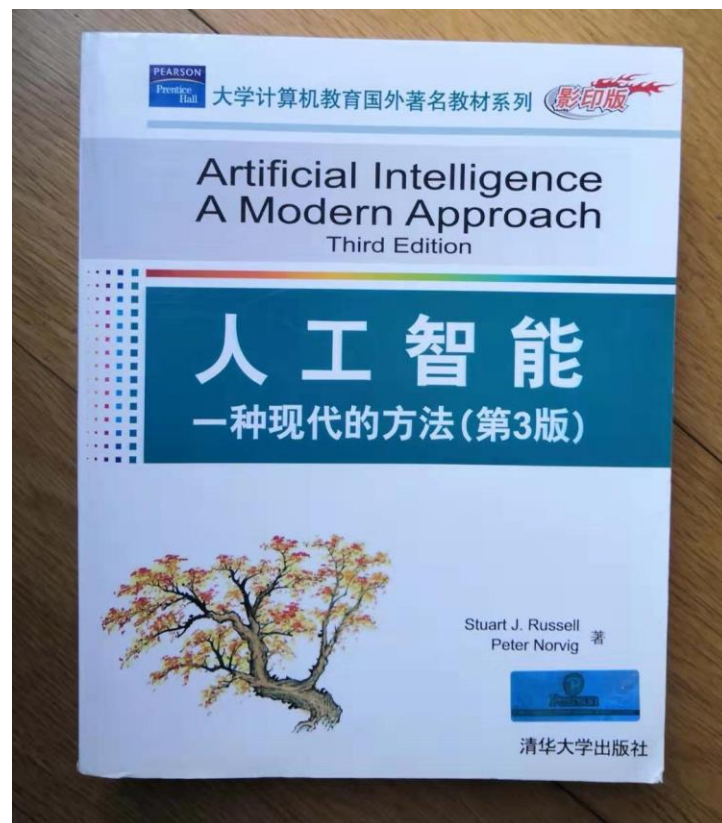
- Overview of the course
- Introduction to AI

# Course information

- 教师：王新杰（金融系助理教授）
  - 答疑时间：提前预约
  - 办公室：慧园3栋320
  - 电话：88018602
  - 邮件：[xinjie.wang@sustech.edu.cn](mailto:xinjie.wang@sustech.edu.cn)（邮件主题加上“FIN311”）
- 助教：陈爱（硕士研究生）
- 课堂时间：
  - 单周1次，双周2次

# Textbook

- Textbook: 人工智能 (第3版, 影印版)  
[美] 拉塞尔 (Stuart J. Russell), [美] 诺维格 (Peter Norvig)  
著 (ISBN: 9787302252955)  
— Jd.com



# AI in Finance

## 智能投顾服务模式示意



# AI in Finance

## Contract Intelligence (COIN) - JPMorgan



### JPMorgan Software Does in Seconds What Took Lawyers 360,000 Hours

By **Hughson**

February 26, 2017, 8:01 AM GMT+0 | Updated on February 26, 2017, 8:04 PM GMT+0

- New software does in seconds what took staff 360,000 hours
- Bank seeking to streamline systems, avoid redundancies

Made possible by investments in machine learning and a new private cloud network, COIN is just the start for the biggest U.S. bank. The firm recently set up technology hubs for teams specializing in big data, robotics and cloud infrastructure to find new sources of revenue, while reducing expenses and risks.

As for COIN, the program has helped JPMorgan cut down on loan-servicing mistakes, most of which stemmed from human error in interpreting 12,000 new wholesale contracts per year, according to its designers.

[20]

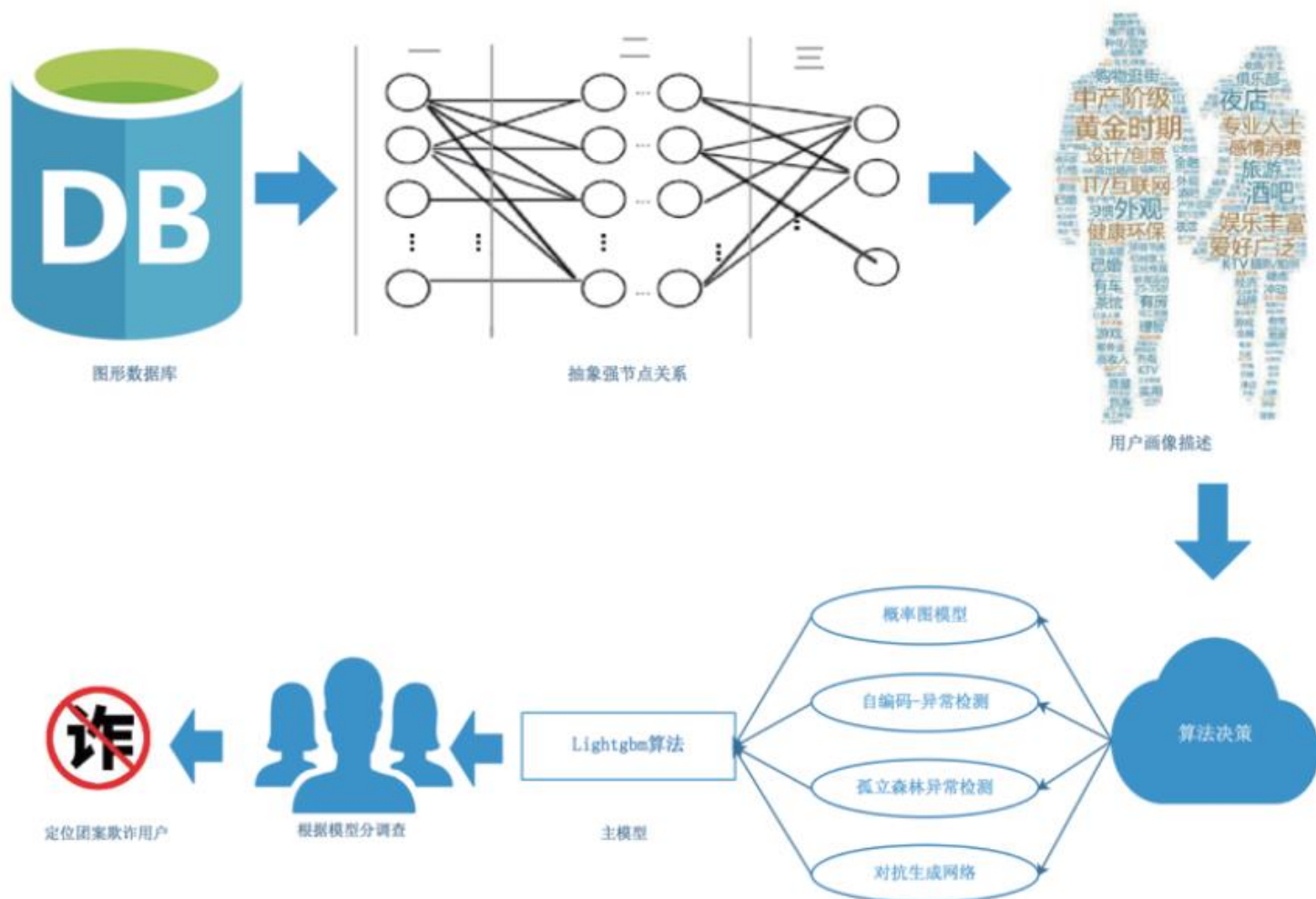
22

# AI in Finance

表1：传统风控和智能风控对比表

传统风控		传统风控
风控模型	以人工审核为主，依靠专家经验	以模型、策略体系自动分析决策为主，人工审核为辅
数据来源	公司内部资料、央行征信资料、客户提交资料	除传统风控数据来源外的第三方数据、线上线下多维度数据
数据维度	数据特征数量少，以基本信息为主的强变量为主	特征数量大于1000，以基本信息、行为特征信息为主的弱变量信息
数据关联性	数据关联度低	数据关联度高，可交叉验证
模型设定	以线性模型为主，因果关系强	以深度学习、集成学习模型为主，可应用相关关系

# AI in Finance





# FinTech

## 金融科技雇主面临的挑战

85%

艰难的过程：  
受访的金融科技雇主表示，  
金融科技人才招聘过程  
充满挑战

本土知识：

60%

大多数受访的金融科技雇主更  
青睐本土培养的人才

39% 的雇主更青睐海归人才



经验不足：



45%

在人才招聘面临的挑  
战中，最关键的一项  
是难以找到符合特定  
职位需求的人才

创新挑战：

金融科技雇主认为，中国金  
融科技人才短缺的一个关键  
原因是

“行业核心技术发展  
太快，导致人才难  
以有效掌握”

金融科技雇主最期待人才具备的技能：



最热门的金融  
科技职位：



40%  
大数据岗位



32%  
人工智能岗位



12%  
风险管理岗位

## Purposes of this course

- Get you started in FinTech (AI)
  - Learn the basic analytical frame for AI
  - Learn basic coding skills for AI
- Start your own AI projects
  - Find an interesting problem
  - Develop an AI solution

## Difference between this course and the similar course from CS

- AI is a huge field.
- This course focuses on AI that is relevant to financial applications
  - Process information
- We will spend on less time on general AI algorithms
- We use Python as programming language

## How to learn this course

- Get your hands dirty!
- Do lots of exercise
- Practice makes perfect!
- Most important of all, you must have ideas and questions!

# How to learn this course

Ideally,

- You spend 70% of time on playing with codes
  - Learn how to use TensorFlow
  - Play with existing codes
- You spend 30% of time on learning principles and thinking
  - Find AI applications already used in Finance
  - Think about new AI applications

# Class requirements

For each lecture,

- Bring your laptop with you
  - We will do programming exercise in class
- Bring a blank paper
  - We will do exercise in class and you will hand in the paper at the end of class.

## Grading

- Attendance: 10%
- Class performance: 10%
- Homework: 30%
- Projects: 50%
- The course grade is based on the amount of work!



FIN311 (2019)

扫一扫二维码，加入该群。



# Programming language

- Python
  - Easy to learn (> C, C++, Java)
  - Widely used in financial industry
  - Rich interfaces to other computer languages
  - Can be used as “calculator” for your career

## Course web

- sakai.sustc.edu.cn
  - FIN311: 人工智能
  - Lecture notes and homework
  - Codes and data
- QQ group
  - Announcement
  - Discussion

# Questions?

## EXERCISE

- It is a good habit to enter into a field with your own idea (even if the idea is wrong). What do you think about AI?

# Lecture 1

## **Introduction**

Artificial intelligence

Python

# Outline

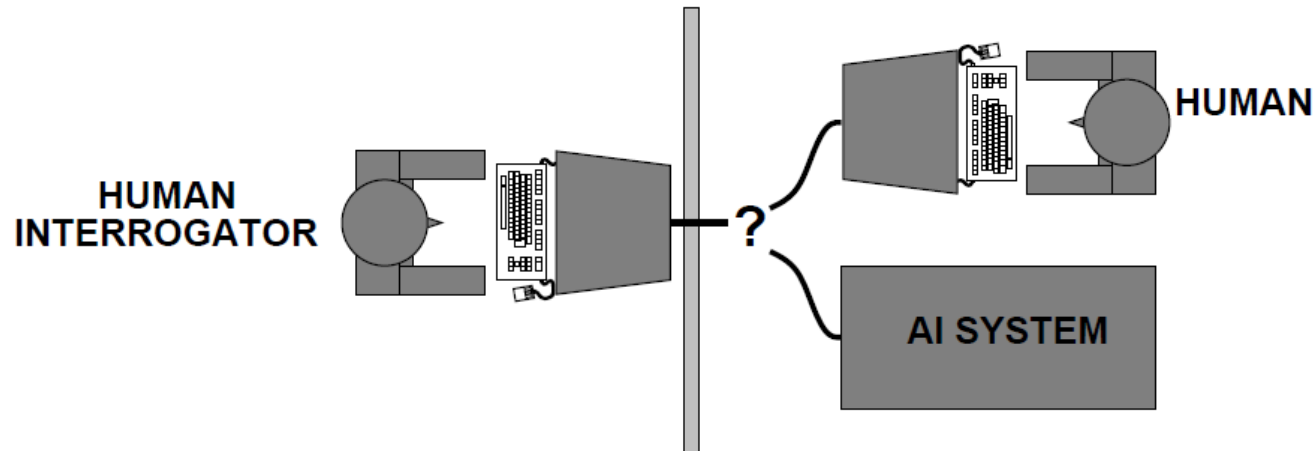
- What is AI?
- A brief history
- The state of the art

# What is AI?

- Systems that think like humans
- Systems that think rationally
- Systems that act like humans
- Systems that act rationally

# Acting humanly: The Turing test

- Turing (1950) "Computing machinery and intelligence":
- "Can machines think?" -> "Can machines behave intelligently?"
- Operational test for intelligent behavior: the Imitation Game



- Predicted that by 2000, a machine might have a 30% chance of fooling a lay person for 5 minutes
- Anticipated all major arguments against AI in following 50 years
- Suggested major components of AI: knowledge, reasoning, language understanding, learning



# Thinking humanly: Cognitive Science

- 1960s “cognitive revolution”: information-processing psychology replaced prevailing orthodoxy of behaviorism
- Requires scientific theories of internal activities of the brain
  - What level of abstraction? “Knowledge” or “circuits”?
  - How to validate? Requires
    - 1) Predicting and testing behavior of human subjects (top-down)
    - or 2) Direct identification from neurological data (bottom-up)
- Both approaches (roughly, Cognitive Science and Cognitive Neuroscience) are now distinct from AI
- Both share with AI the following characteristic:
  - the available theories do not explain (or engender)
  - anything resembling human-level general intelligence
- Hence, all three fields share one principal direction!

# Thinking rationally: Laws of Thought

- Normative (or prescriptive) rather than descriptive
- Aristotle: what are correct arguments/thought processes?
- Several Greek schools developed various forms of logic:
  - notation and rules of derivation for thoughts;
- may or may not have proceeded to the idea of mechanization
- Direct line through mathematics and philosophy to modern AI

## Problems:

- 1) Not all intelligent behavior is mediated by logical deliberation
- 2) What is the purpose of thinking? What thoughts should I have out of all the thoughts (logical or otherwise) that I could have?

# Acting rationally

- Rational behavior: doing the right thing
- The right thing: that which is expected to maximize goal achievement, given the available information
- Doesn't necessarily involve thinking - e.g., blinking reflex | but thinking should be in the service of rational action

- Aristotle (Nicomachean Ethics):

Every art and every inquiry, and similarly every action and pursuit, is thought to aim at some good

# Rational agents

- An agent is an entity that perceives and acts
- This course is about designing rational agents
- Abstractly, an agent is a function from percept histories to actions:

$$f : \mathcal{P}^* \rightarrow \mathcal{A}$$

- For any given class of environments and tasks, we seek the agent (or class of agents) with the best performance
- Caveat: computational limitations make perfect rationality unachievable
- design best program for given machine resources

# AI prehistory

- Philosophy
  - logic, methods of reasoning
  - mind as physical system
  - foundations of learning, language, rationality
- Mathematics
  - formal representation and proof
  - algorithms, computation, (un)decidability, (in)tractability
  - probability
- Psychology
  - Adaptation
  - phenomena of perception and motor control
  - experimental techniques (psychophysics, etc.)
- Economics
  - formal theory of rational decisions
- Linguistics
  - knowledge representation
  - grammar
- Neuroscience
  - plastic physical substrate for mental activity
- Control theory homeostatic systems, stability
  - simple optimal agent designs

# Potted history of AI

- 1943 McCulloch & Pitts: Boolean circuit model of brain
- 1950 Turing's "Computing Machinery and Intelligence"
- 1952-69 Look, Ma, no hands!
- 1950s Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine
- 1956 Dartmouth meeting: "Artificial Intelligence" adopted
- 1965 Robinson's complete algorithm for logical reasoning
- 1966-74 AI discovers computational complexity  
Neural network research almost disappears
- 1969-79 Early development of knowledge-based systems
- 1980-88 Expert systems industry booms
- 1988-93 Expert systems industry busts: "AI Winter"
- 1985-95 Neural networks return to popularity
- 1988- Resurgence of probability; general increase in technical depth  
"Nouvelle AI": ALife, GAs, soft computing
- 1995- Agents, agents, everywhere...
- 2003- Human-level AI back on the agenda

# State of the art

Which of the following can be done at present?

- Play a decent game of table tennis
- Drive safely along a curving mountain road
- Drive safely along Telegraph Avenue
- Buy a week's worth of groceries on the web
- Buy a week's worth of groceries at Berkeley Bowl
- Play a decent game of bridge
- Discover and prove a new mathematical theorem
- Design and execute a research program in molecular biology
- Write an intentionally funny story
- Give competent legal advice in a specialized area of law
- Translate spoken English into spoken Swedish in real time
- Converse successfully with another person for an hour
- Perform a complex surgical operation
- Unload any dishwasher and put everything away

# State of the art





Python

# Python

- Python is a programming language that lets you work quickly and integrate systems more effectively.

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```



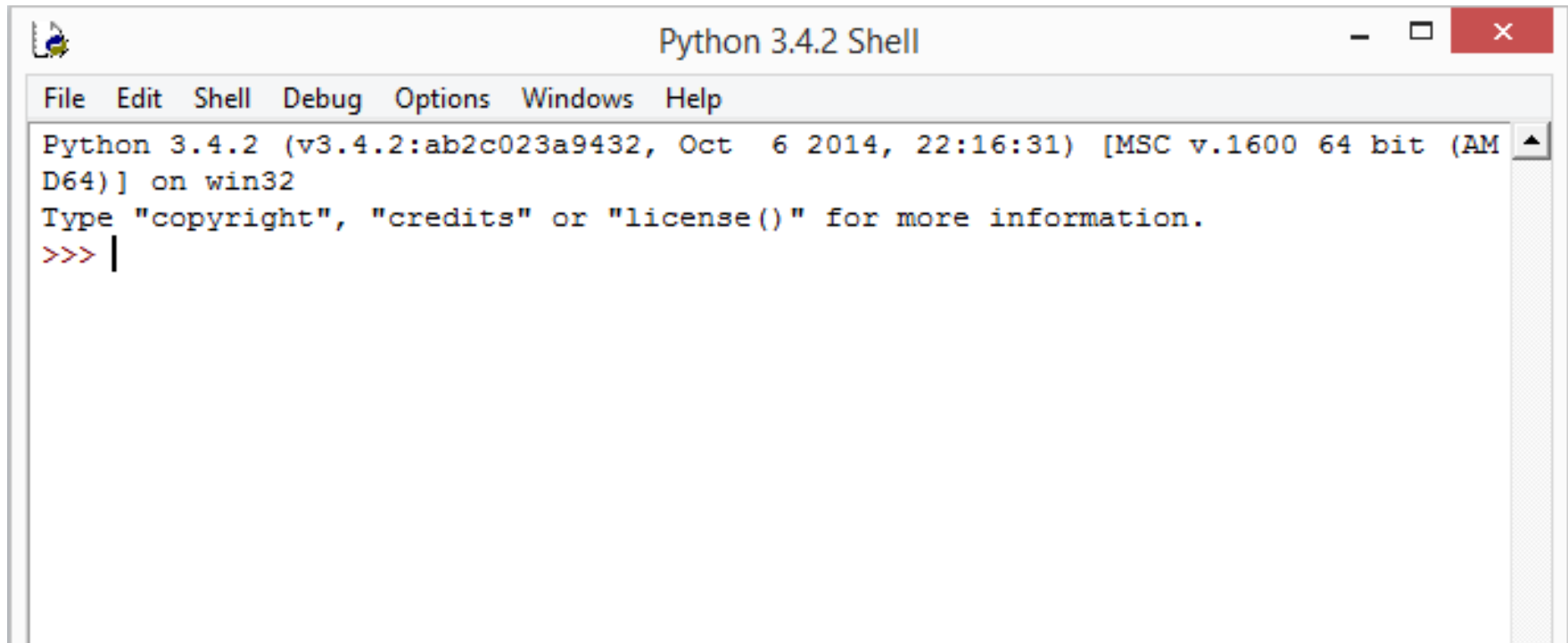
# Install Python

Python can run on many operating systems.

For windows:

1. <https://www.python.org/downloads/release/python-372/>
2. Click “Windows x86-64 executable installer”
3. I also uploaded the installer on Sakai. You can download it from there.

# Python command line

A screenshot of a Windows application window titled "Python 3.4.2 Shell". The window has a standard Windows interface with a title bar, a menu bar (File, Edit, Shell, Debug, Options, Windows, Help), and a main text area. The text area displays the Python 3.4.2 startup message: "Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:16:31) [MSC v.1600 64 bit (AMD64)] on win32". Below this, it says "Type 'copyright', 'credits' or 'license()' for more information." and then the prompt ">>> |" is shown, indicating the shell is ready for user input. The window has standard minimize, maximize, and close buttons in the top right corner.

```
Python 3.4.2 Shell
```

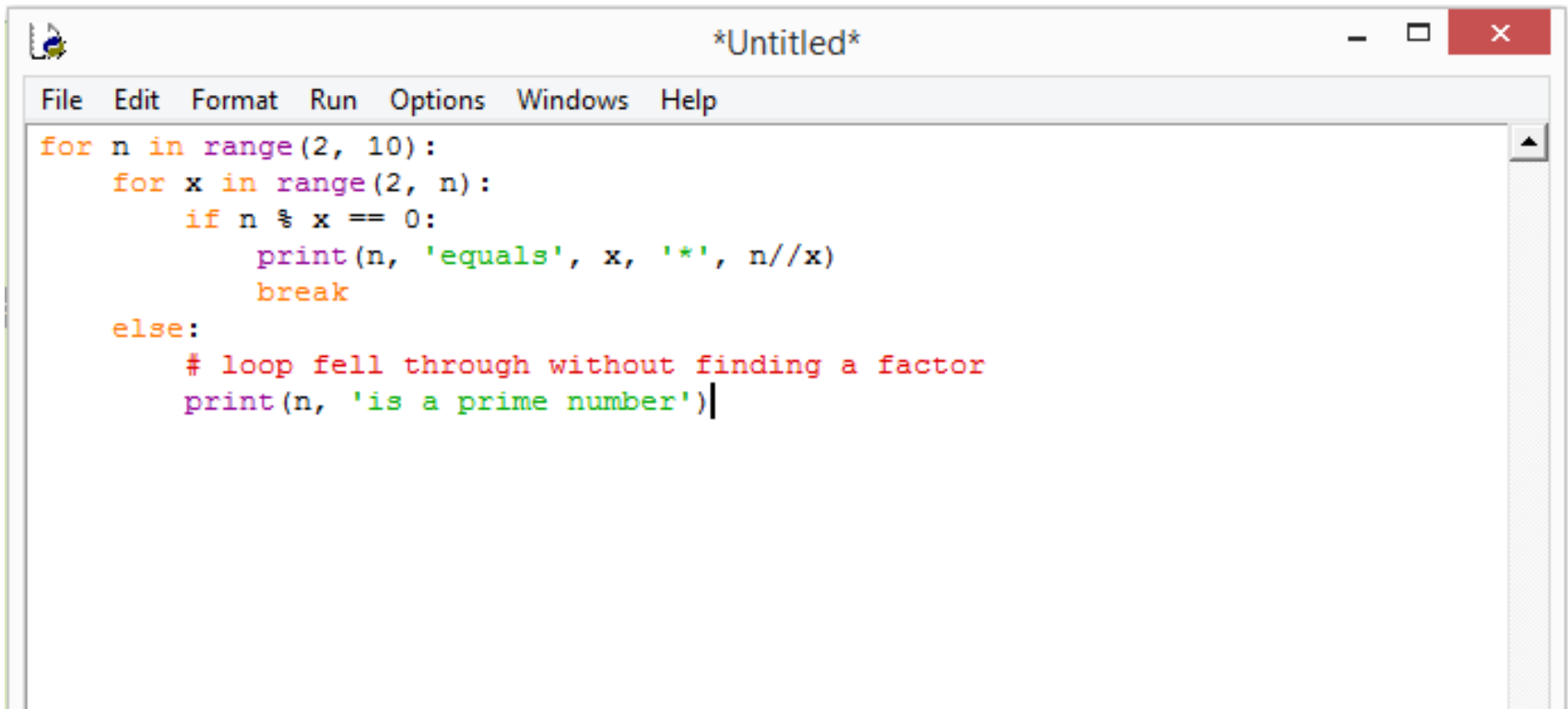
```
File Edit Shell Debug Options Windows Help
```

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:16:31) [MSC v.1600 64 bit (AMD64)] on win32
```

```
Type "copyright", "credits" or "license()" for more information.
```

```
>>> |
```

# Python script



```
for n in range(2, 10):
    for x in range(2, n):
        if n % x == 0:
            print(n, 'equals', x, '*', n//x)
            break
    else:
        # loop fell through without finding a factor
        print(n, 'is a prime number')
```

# Using Python as a Calculator

```
>>> 2 + 2
```

```
4
```

```
>>> 50 - 5*6
```

```
20
```

```
>>> (50 - 5*6) / 4
```

```
5.0
```

```
>>> 8 / 5
```

```
1.6
```

```
>>> width = 20
```

```
>>> height = 5 * 9
```

```
>>> width * height
```

```
900
```

# Using Python as a Calculator

```
>>> prefix = 'Py'
>>> prefix + 'thon'
'Python'
>>> word = 'Python'
>>> word[0] # character in position 0
'P'
>>> word[5] # character in position 5
'n'
```

# If

```
>>> x = int(input("Please enter an integer: "))
Please enter an integer: 42
>>> if x < 0:
...     x = 0
...     print('Negative changed to zero')
... elif x == 0:
...     print('Zero')
... elif x == 1:
...     print('Single')
... else:
...     print('More')
...
More
```



# For

```
>>> # Measure some strings:
... words = ['cat', 'window', 'defenestrate']
>>> for w in words:
...     print(w, len(w))
...
cat 3
window 6
defenestrate 12
```

# Range()

```
>>> for i in range(5):  
...     print(i)  
...  
0 1 2 3 4  
range(5, 10)  
5, 6, 7, 8, 9  
range(0, 10, 3)  
0, 3, 6, 9  
range(-10, -100, -30)  
-10, -40, -70
```

# Define a function

```
>>> def fib(n):      # write Fibonacci series up to n
...     """Print a Fibonacci series up to n."""
...     a, b = 0, 1
...     while a < n:
...         print(a, end=' ')
...         a, b = b, a+b
...     print()
...
>>> # Now call the function we just defined:
... fib(2000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597
```

## EXERCISE

- Write a Python code that calculates  $\sum_{i=1}^{10} i^2$ .

# HOMEWORK

- Install Python and packages on your laptop
- Run all Python commands in this lecture.