

---

PAMUKKALE ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ  
2021 BAHAR

# Biçimsel Diller ve Otomata Teorisi

Formal languages  
and automata theory

---

# Bilgisayarların Yetenekleri

立法會譚偉豪議員對財政預算案的回應 Inbox | X

☆ Office of Hon. Samson Tam office@samsontam.hk to Andrej show details Feb 23 Reply

Chinese (Traditional Han) > English **Translate message** Turn off for: Chinese (Traditional Han)



OFFICE OF THE HON. SAMSON TAM (Information Technology)  
立法會譚偉豪議員辦事處 (資訊科技界)

## 對2011-12年度財政預算案的回應

財政司司長今天發表的新一份財政預算案，以下是我對預算案的初步回應。

對於新一份財政預算案，我主要有三方面回應：

1. 在「十二·五」規劃下，特區政府在爭取香港新位上有欠清晰，表現不夠進取；
2. 歡迎政府將「創新及科技督導委員會」升格，期望在財政司司長領導下，政府會透過實質撥款和制訂具體政策，並成立創新及科技局，以推動創新科技產業發展；
3. 期望政府將發展香港成為高端數據中心，列作「十二·五」規劃下的重點爭取項目。

首先，我認為，經濟發展對香港有著重要影響，但現時本港經濟仍然側重於金融及地產，經濟發展欠多元化，實在是不健康。香港經濟要有持續的健康發展，必須把握「十二·五」規劃的契機，盡快為香港定下明確的新定位，不過，我覺得，現階段政府在這方面所做的，實在不夠進取，令人有點失望。



立法會譚偉豪議員對財政預算案的回應 (Legislative Council Hon.'s Response to the Budget) Inbox | X

☆ Office of Hon. Samson Tam office@samsontam.hk to Andrej show details Feb 23 Reply

Chinese (Traditional Han) > English View original message Always translate: Chinese (Traditional Han)



OFFICE OF THE HON. SAMSON TAM (Information Technology)  
立法會譚偉豪議員辦事處 (資訊科技界)

## Budget for 2011-12 in response

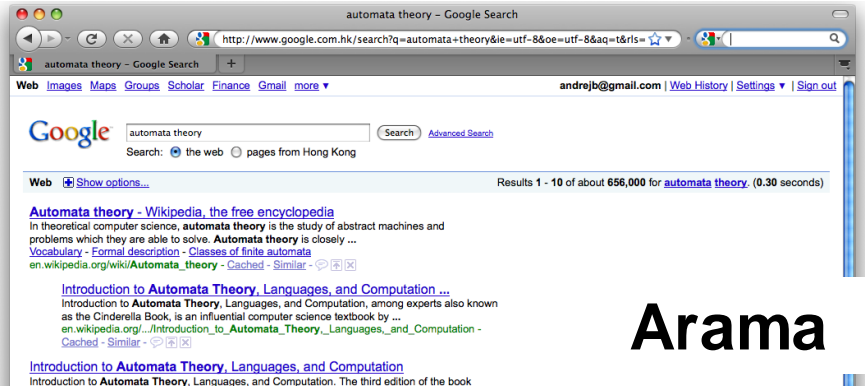
The Financial Secretary today's new Budget, the following is my initial response to the Budget.

For the new Budget, I have three main responses:

- 1 in the "second · Five Year Plan", the Hong Kong SAR Government for lack of clarity on the new position, performance is not aggressive;
  - (2) welcomes the Government will be "Innovation and Technology Steering Committee" upgraded, hoping that the Financial Secretary under the leadership of the Government, through substantial funding and to develop specific policies and Innovation and Technology to promote innovation and technology development;
  3. Expect the government to develop Hong Kong into a high-end data center, classified as "second · Five-Year" plan for the project under the key.
- First of all, I think, economic development has an important impact on Hong Kong, but now the economy is still focused on the financial and real estate, less diversified economic development, it is unhealthy. Hong Kong's economy should be sustained and healthy development, we must grasp the "second · Five Year Plan" an opportunity for Hong Kong as soon as possible to set clear new position, but I think at this stage did the government in this regard, it is not aggressive, somewhat



# Başka Neler?



Arama

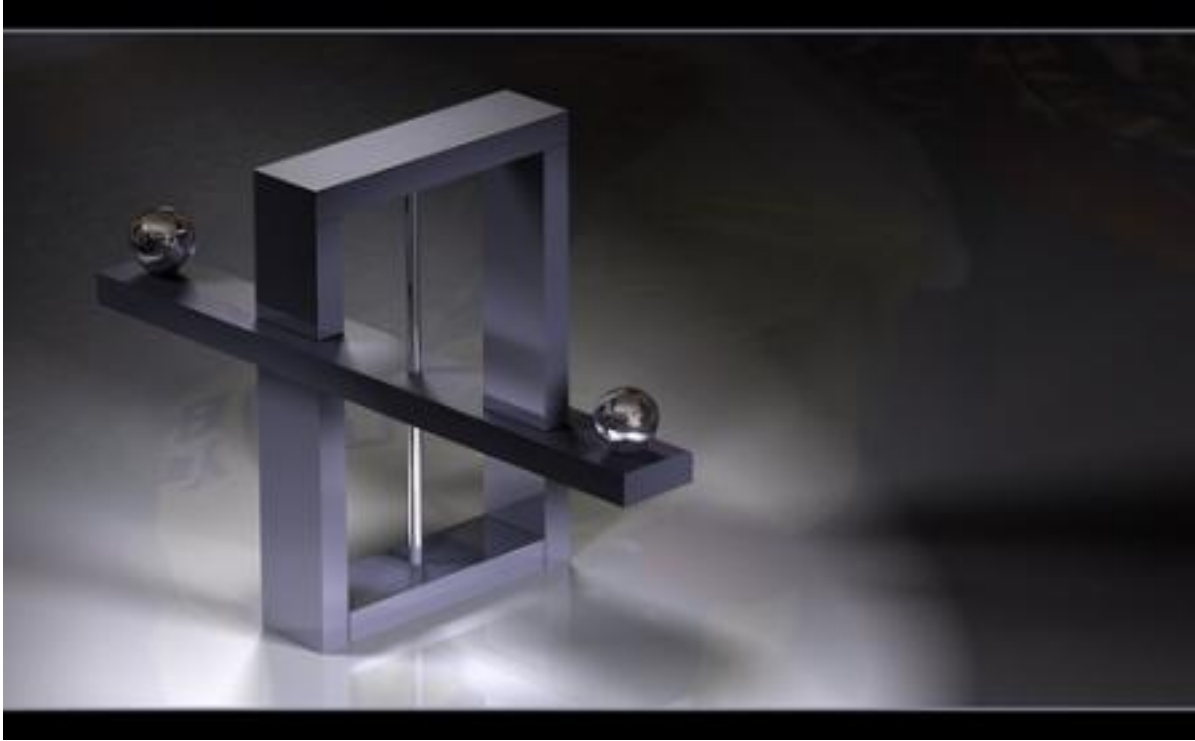


Otomatik  
Uçuş

Bir bilgisayarın yapamayacağı bir şey var mı?

# İmkansızlık

---



İmkansız olan durumları neden  
önemsiyoruz?

# Ebedi Hareket (Perpetual motion)



Orta çağlarda insanlar hiç enerji kullanmayan bir makine istiyorlardı.

Daha sonra fizikteki keşifler, enerjinin yoktan var edilemeyeceğini gösterdi.

Perpetual motion → boş çaba

İmkansızı anlamak, enerjimizi daha yararlı olana yönlendirmemize yardımcı olur.

# Hesaplama Yasaları

---

Tıpkı fizik yasaları bize doğada olması mümkün olan veya olmayanları söylemesi gibi ...

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \quad \nabla \cdot \mathbf{B} = 0 \quad \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \quad \nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$

... hesaplama yasaları bize bilgisayarlar için neyin mümkün olduğunu veya olmadığını söyler.

# Hesaplanabilirlik (Computability)

---

- Hesaplanabilirlik, hesaplama probleminin bir özelliğidir. Daha özel olarak, **sınırlı zamanda sınırsız belleğe sahip bir bilgisayar tarafından gerçekleştirilebilen problemi çözen bir algoritma varsa, bir problem hesaplanabilirdir.** “Sonlu zamanın” üst sınırının ne olduğu önemli değil; bu sorunu çözen bir algoritmayı çalıştırmak için üssel olarak çok sayıda adım veya keyfi olarak büyük miktarda bellek gerekebilir, ancak hesaplanabilirse, algoritmanın sınırlı bir süre içinde sonuçlanacağını biliyoruz. Bununla birlikte, neyin hesaplanabilir olması gerektiğine ilişkin bu sınırlandırılmamış bir sınırla bile, hala hesaplanamayan, yani bunları çözmek için böyle bir algoritmanın olmadığını kanıtlayabileceğimiz sorunlar vardır.

# Otomata Teorisi

---

Otomata teorisi hesaplama yasalarını inceler.

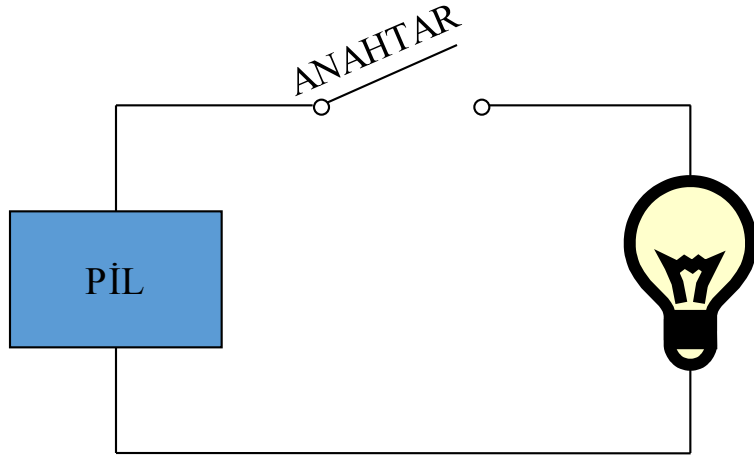
- Otomata teorisi, soyut hesaplama cihazlarının incelenmesidir
- Soyut cihazlar, gerçek hesaplamaların (basitleştirilmiş) modelleridir
- Hesaplamalar her yerde olur: Dizüstü bilgisayarınızda, cep telefonunuzda, doğanın içinde ...
- Neden soyut modellere ihtiyacımız var?

Gerçekte, hesaplama yasaları tam olarak anlaşılmamıştır, ancak otomata teorisi iyi bir başlangıçtır.



# Basit bir bilgisayar

---



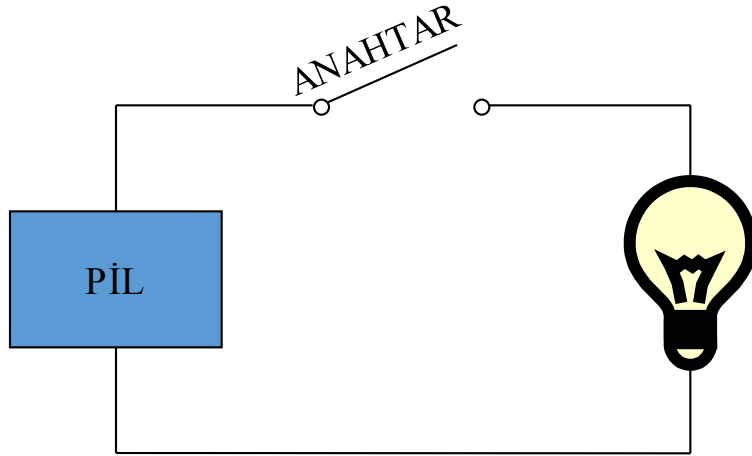
**giriş:** anahtar

**çıkış:** lamba

**eylem:** anahtarın çevrilmesi

**states:** açık/kapalı on/off

# Basit bir bilgisayar

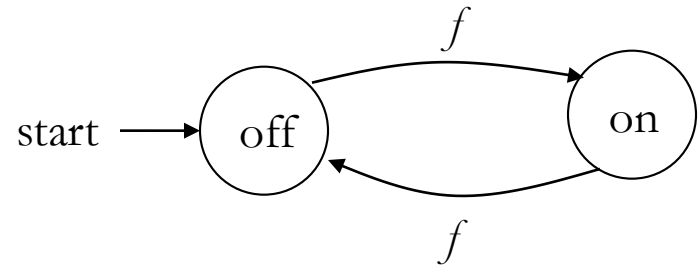


**giriş:** anahtar

**çıkış:** lamba

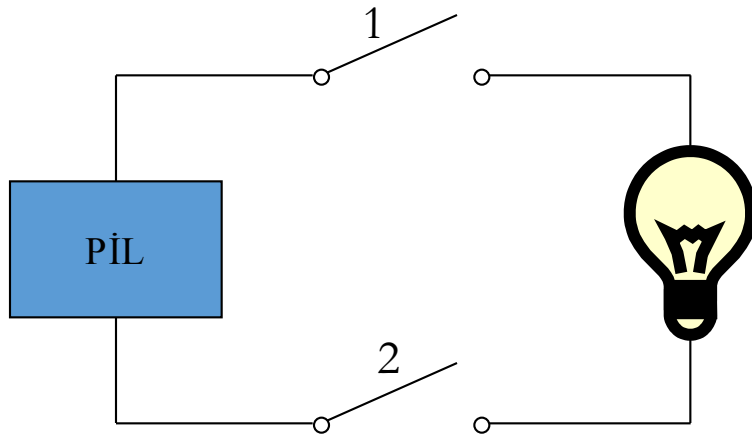
**eylem:**  $f$  (anahtarın çevrilmesini ifade eden eylem)

**states:** *on/off*



ampul, ancak ve ancak  
tek sayıda çevirme varsa  
yanar (başlangıç: off)

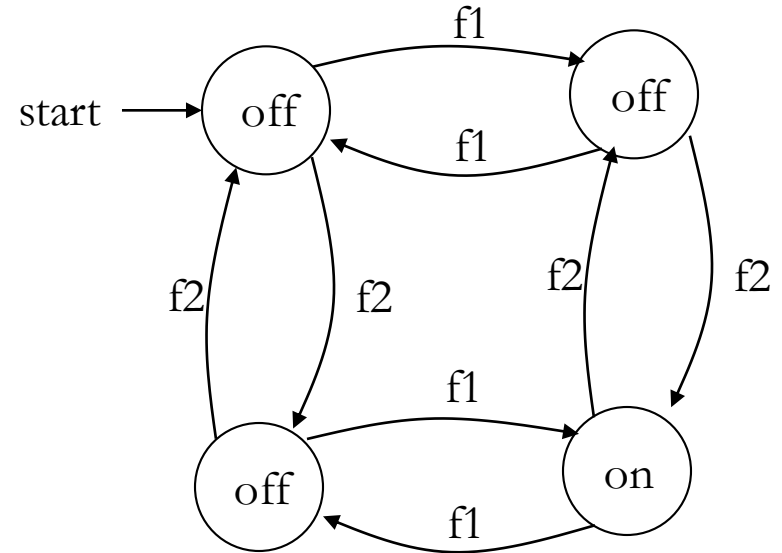
# Başka bir “bilgisayar”



**inputs:** anahtar 1 ve 2

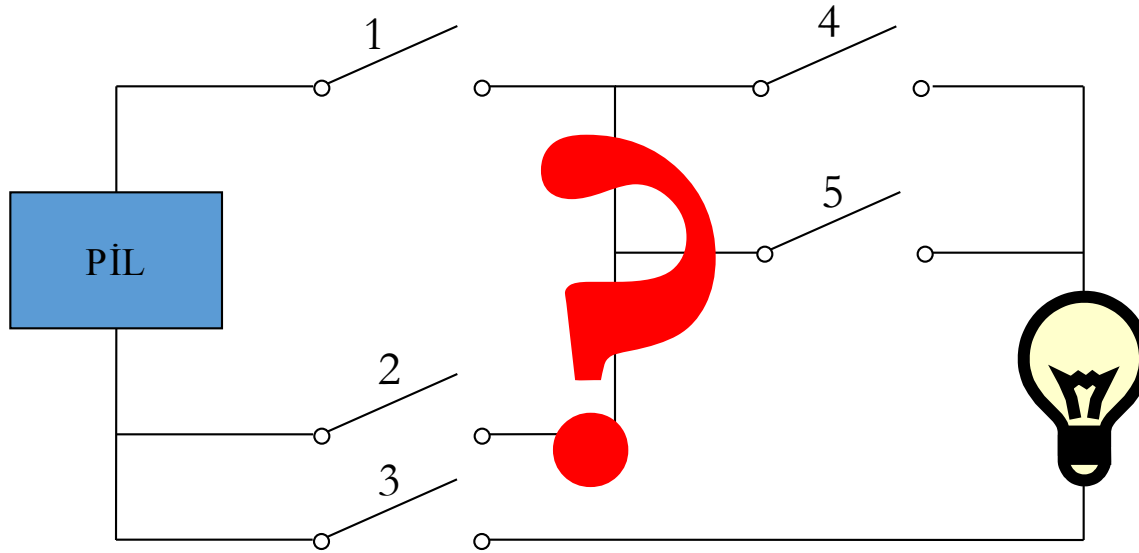
**actions:** f1 “anahtar 1 için”  
f2 “anahtar 2 için”

**states:** on, off



ampul ancak ve ancak her iki düğme de tek sayıda çevrildiğinde yanar.

# Tasarım Problemi



Yalnızca ve ancak **tüm anahtarlar tam olarak aynı sayıda çevrildiğinde** ışığın açık olduğu bir devre tasarlayabilir misiniz?

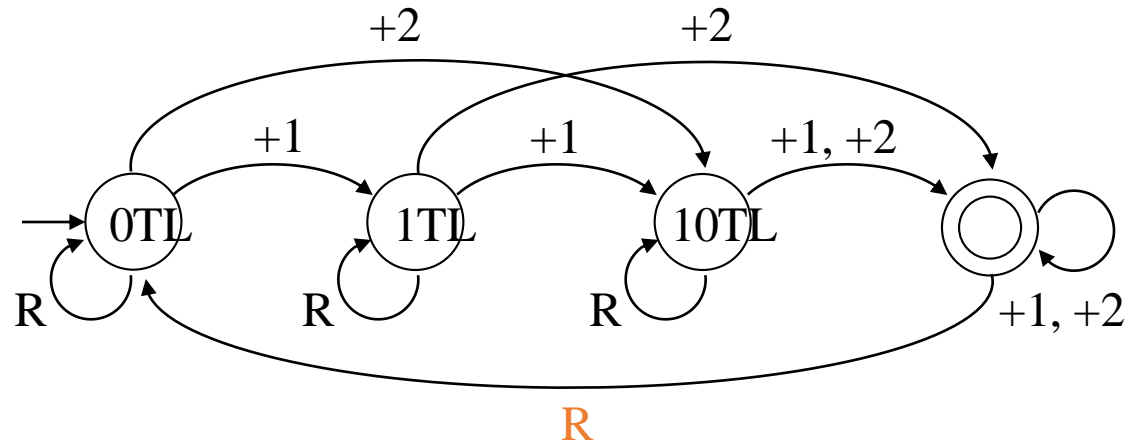
# Bir sakız makinesi



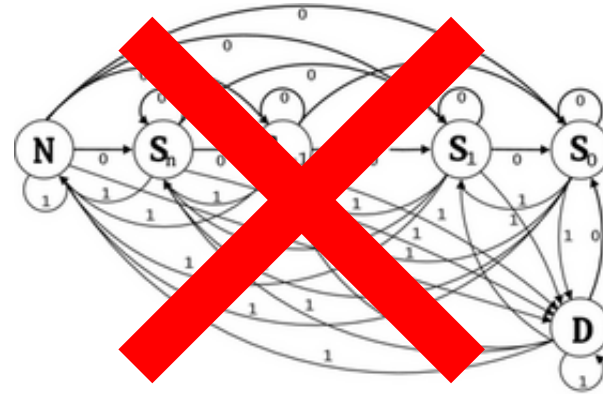
Makine 1TL ve 2TL ile bozuk ile çalışsın.

Bir sakız 3 TL olsun

eylemler: +1 at, +2 at, çevir (R)



# Slot makinesi

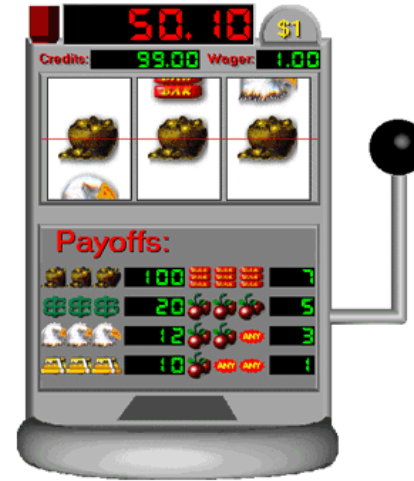


=



# What is the difference?

---



- Bu tür soruların akıl yürütmesi zordur, çünkü cihazlar sonsuz sayıda farklı şekilde tasarlanabilir.
- Otomata Teorisi bu soruların bazılarını cevaplamamızı sağlar.

# Otomata ne yapabilir

---

- Küçük bir cihazın çalışmasını tanımlayabilirler
- (Basit) yazılımı doğrulamak için kullanılabilirler \*
- Sözcük çözümleyicilerinde (lexical analyzer) programlama dillerindeki ifadeleri tanımak için kullanılırlar:

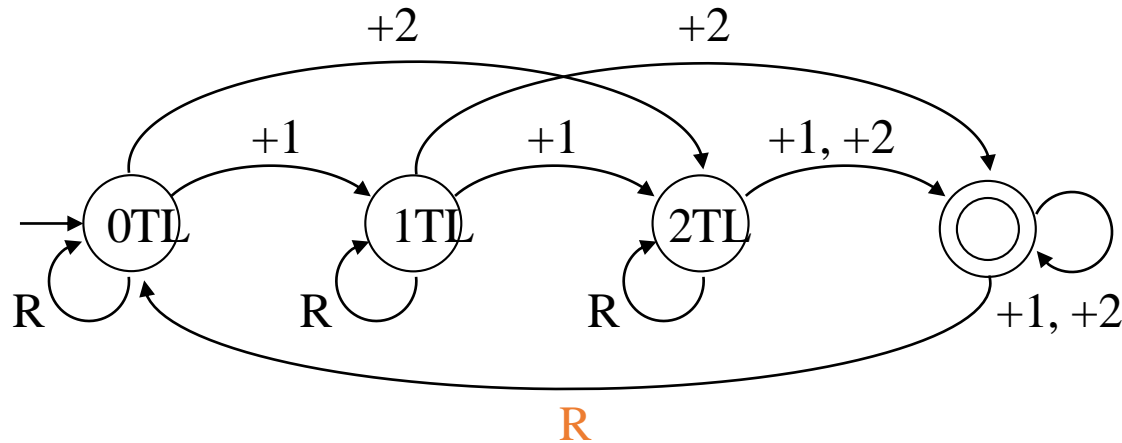
ab1 is a legal name of a variable in Java

5u= is not



# Farklı tür makineler

---



- Bu, bir makinenin yalnızca bir örneği
- Farklı türdeki makinelere bakıp soracağız:
  - Bu tür bir makine ne tür sorunları çözebilir?
  - Bu tür bir makine için imkansız şeyler nelerdir?
  - Makine A, Makinesi B'den daha mı güçlü?

# Bazı tür makineler

---

---

## finite automata

Az miktarda belleğe sahip cihazlar. Çok basit şeyleri modellemek için kullanılır.

---

## push-down automata

Sınırlı bir şekilde erişilebilen sonsuz belleğe sahip cihazlar.

Grammer ayrıştırma (Parse Grammar) için kullanılır

---

## Turing Machines

Sonsuz hafızalı cihazlar.

Bunlar gerçek bilgisayarlar

---

## time-bounded Turing Machines

Sonsuz bellek, ancak sınırlı çalışma süresi.

Bunlar oldukça hızlı çalışan bilgisayarlardır.

---

# Dersin bazı önemli noktaları

---

- **Sonlu Otomata (Finite automata)**

- Otomatlar, metindeki kalıpları arama göreviyle yakından ilgilidir

`find (ab)* (ab) in abracadabra`

- **Gramer (Grammars)**

- Gramer, konuşma dilindeki cümlelerin anlamını ve bilgisayar programlarının anlamını tanımlar.
- Bir bilgisayar programından anlamın çıkarılmasını gramer kuralları sağlar (formal languages)

# Dersin bazı önemli noktaları

---

- Turing Machines
  - Bu, hesaplamayı umduğumuz her şeyi başaran genel bir bilgisayar modelidir.
  - Ancak bilgisayarların yapamayacağı birçok şey var:

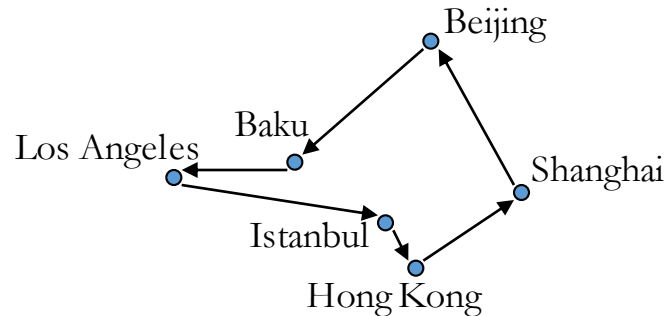
## The halting problem

Bir programlama dersinde özyinelemeyi ilk kez öğrenirken, kazara sonsuza kadar yinelenen bir kod yazmış ve kötü şöhretli bir "yığın taşması" hatası oluşturmuş olabilirsiniz. Bunun kodu çalıştırmadan gerçekleşeceğini bilmenin bir yolu olmasını dilemiş olabilirsiniz.

# Dersin bazı önemli noktaları

---

- Time-bounded Turing Machines
  - Prensipde bir bilgisayarda birçok problem çözmek mümkündür, ancak pratikte çok fazla zaman alır
  - Gezgin Satıcı Problemi (TSP): Bir şehir listesi verildiğinde, onları bir defa ziyaret etmenin ve eve geri dönmenin en kısa yolunun bulunması.



- Pratikte zor: 100 şehir için bu, en hızlı bilgisayarda bile 100+ yıl sürecektir!

# Otomata Teorisinin Temeli

---

- Soruyu nasıl sorarız

**A makinesi B problemini çözebilir mi?**

- Öncelikle, çözmek istediğimiz sorunları tanımlamanın bir yoluna ihtiyacımız var

# Problemler

---

- Dikkate alacağımız sorun örnekleri
- Bir  $s$  kelimesi verildiğinde, alt kelime olarak «be» içeriyor mu?
- Bir  $n$  sayısı verildiğinde, 7'ye bölünebilir mi?
- İki kelime  $s$  ve  $t$  verildiğinde, bunlar aynı mı?
- Bunların hepsinin "evet / hayır" yanıtları var.
- "Bunu bul" veya "Kaç tane" gibi başka tür sorunlar da vardır, ancak bunlara bakmayacağız.

- 
- Ayrık matematik
  - Belirli bir alana gerek yoktur, ancak ders doğası gereği matematikseldir.
  - Tümevarım yoluyla fonksiyonlar ve ilişkiler, kümeler veya ispatlar hakkında giriş dersinde ele alacağız.
  - Ayrık Matematik ve Veri Yapıları dersleri bu ders için yapılan çalışmaları desteklemektedir.



# Ders Değerlendirme Kriterleri

---

- Vize : %40
- Final: %60
- Ödevler: extra credit
- Derse Katılım: öğrencinin sorumluluğunda

# Referenslar

---

- Notes will be provided for every lecture at EDS (education support system).

The following references cover some of the topics in more detail.

Can be obtained from      Library Genesis   libgen.is

## MAIN BOOK:

- **Elements of the Theory of Computation**, by **Lewis** and **Papadimitrou**, second edition, 1998.

## SUPPLEMENTARY:

- *Introduction to the Theory of Computation*, second edition, by Michael **Sipser**
- *Introduction to Automata Theory, Languages, and Computation*, 3rd edition, by John **Hopcroft**, Rajeev Motwani, and Jeffrey Ullman
- Elements of Computation Theory, **Arindrama Singh**
- *Theory of Computing. A Gentle Introduction*, Efim Kinber & Carl Smith
- Video lectures by [Shai Simonson](#).
- [www.class-central.com](http://www.class-central.com) Stanford University Automata Theory Course, Video lectures from the great [Jeffrey Ullman](#)
- [GitHub - yurifw/Visual-Automata-Simulator](#) by Jean Bovet
- <https://www2.cs.duke.edu/csed/jflap/> form Duke University

# Ders İçeriği

---

- **Introduction**
  - Presentation of the Course
  - Repetition of Data Structures and Basic Discrete Mathematics Concepts
  - Computational Problems and Solvability
- **Regular Languages and Finite Automata**
  - Deterministic Finite Automata (DFA – DSO)
  - Minimization of Finite Automata
  - Nondeterministic Finite Automata (NFA -NDSO)
  - Determinization of Automata
  - Regular Expressions and Languages (RE –RL)
  - Restrictions of Regular Languages: the Pumping Lemma (PL)
- **Context-free Languages and Pushdown Automata**
  - Context-free Grammars and Languages
  - The Parsing Problem for Grammars
  - Pushdown Automata (PDA)
  - Restrictions of Context-free Languages
- **Introduction to Theory of Computation**
  - Unrestricted grammars
  - Turing Machines (TM)
  - Decidability