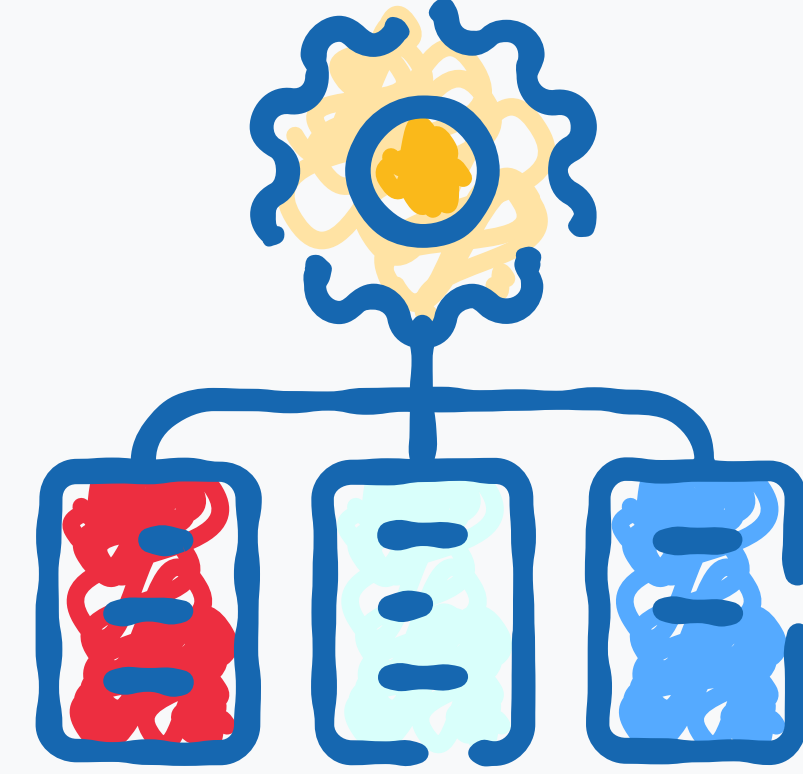


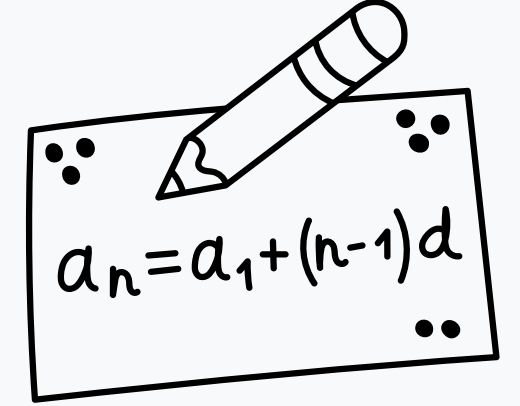
ROBOTİK KODLAMADA VERİ SAKLAMA

Bu Hafta Ne İşleyeceğiz?

Şimdiye kadar robotlarımıza ne yapacaklarını algoritmalarla adım adım anlattık ve test ettik. Ancak robotların, tıpkı bizim gibi bir "hafızaya" ihtiyacı vardır. Bir robot, topladığı elmaları sayarken veya sensöründen gelen sıcaklık değerini okurken bu bilgileri nerede aklında tutar? İşte bugün, kodlama dünyasının en temel yapı taşlarından ikisiyle, yani robotlarımızın hafıza kutuları olan Değişkenler ve Sabitler ile tanışıyoruz!



Değişken ve Sabitler

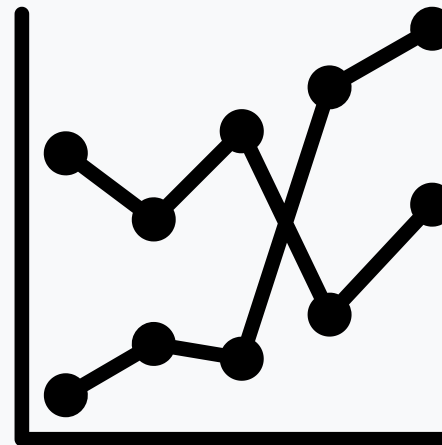


Değişken (Variable) Nedir?

Tanım: Program çalıştığı sürece içinde farklı değerler (sayı, kelime vb.) saklayabilen ve içindeki değerlerin zamanla değişebildiği hafıza alanlarına değişken denir.

Nasıl Hayal Etmeliyiz?: Değişkeni, üzerine etiket yapıştırılmış boş bir karton kutu gibi düşünebilirsiniz. Kutunun adı (etiketi) hep aynı kalır, ancak içine koyduğunuz eşya sürekli değişebilir.

Örnek: Üzerinde "Puan" yazan bir kutumuz olsun. Oyuna başladığımızda kutunun içinde 0 vardır. Altın topladığımızda içindeki 0'ı çöpe atıp yerine 10 koyarız. Kutunun adı hala "Puan"dır ama içindeki değer değişmiştir.

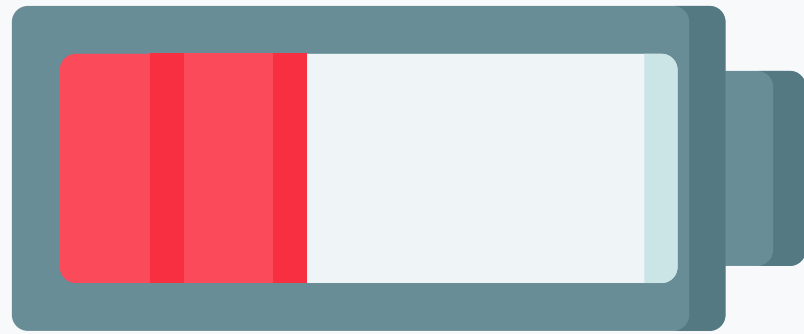


Değişken ve Sabitler

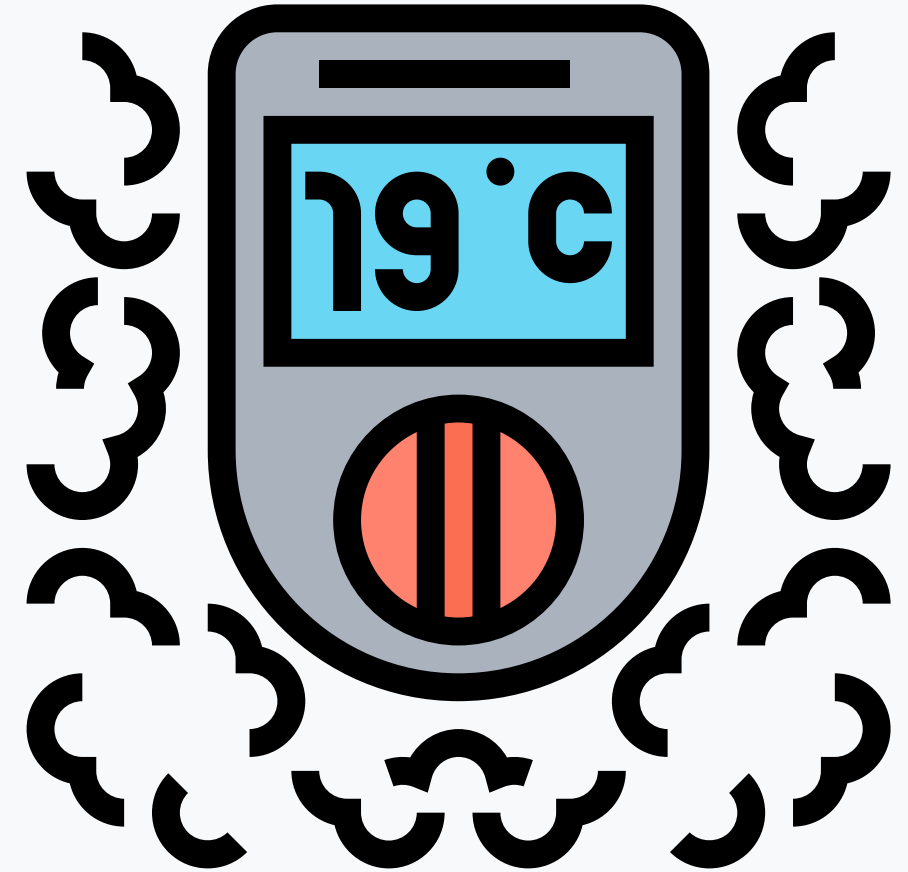
Robotik Sistemlerde Değişken Örnekleri

Robotlar çevreden aldıkları anlık verileri değişkenlerde saklarlar:

- **Oda_Sıcaklığı:** Sensör her saniye yeni bir değer okur (Önce 22, sonra 23, sonra 24 derece).
- **Kalan_Pil:** Robot çalıştıkça pil yüzdesi sürekli azalır (%100, %99, %98).
- **Mesafe:** Robot duvara yaklaştıkça ölçtüğü mesafe değeri anlık olarak değişir.

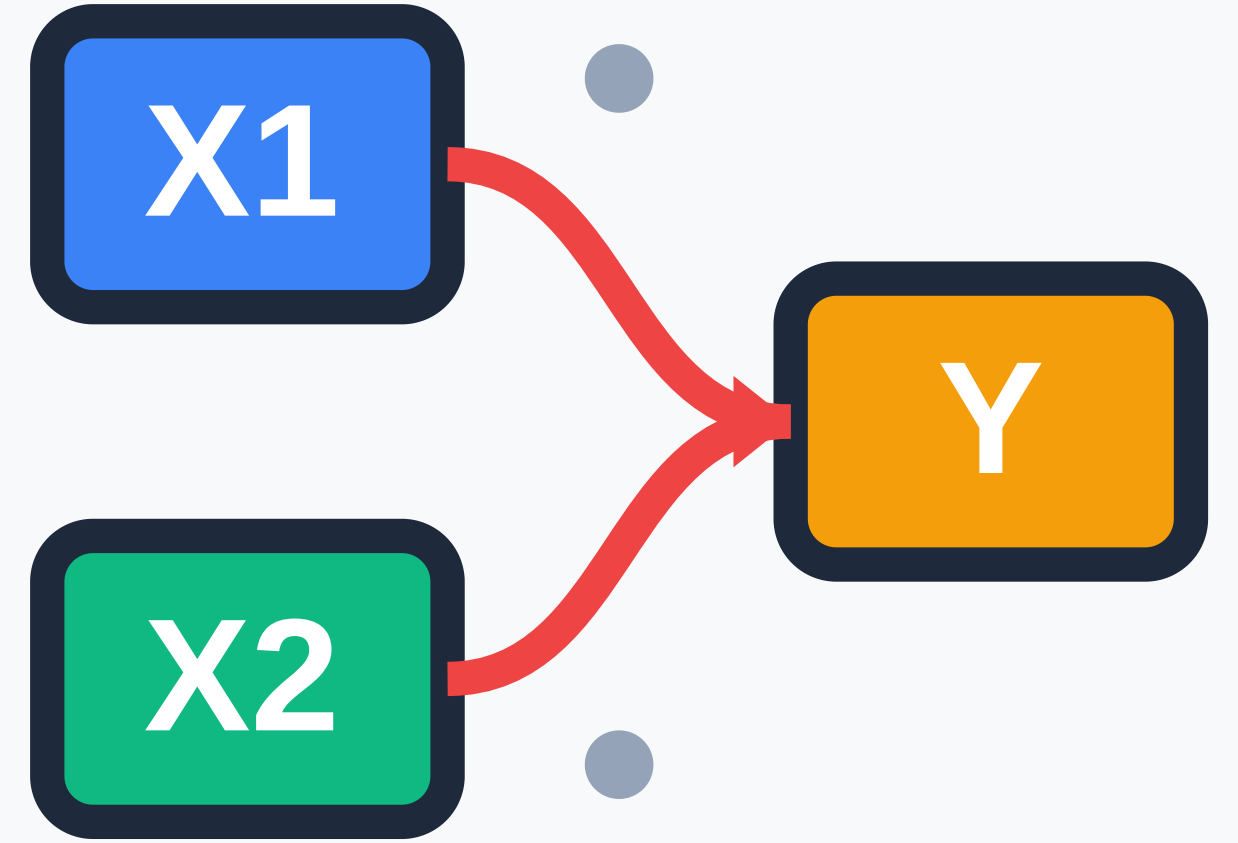


20 MIN.

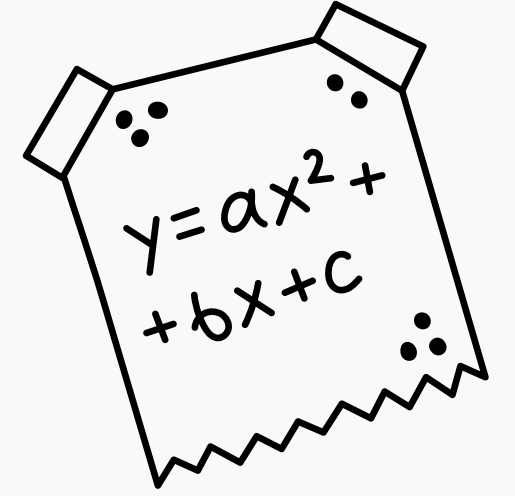


UNUTMA!

Değişkenlere isim verirken içindeki bilgiye uygun, mantıklı isimler seçmeliyiz. Puanları tutacağımız kutunun adını "Elma" koyarsak, kodlar karmaşıklaştığında kafamız karışır. İyi bir bilişimci, değişkenlerine her zaman ne işe yaradığını anlatan kısa ve net isimler verir (Örn: toplam_puan, kalan_can).



Değişken ve Sabitler



Sabit (Constant) Nedir?

Tanım: Program çalıştığı sürece içindeki değeri asla değiştirilemeyen, ilk baştan nasıl belirlendiyse hep öyle kalan veri saklama alanlarına sabit denir.

Nasıl Hayal Etmeliyiz?: Sabitleri, şifresini sadece en başta belirlediğiniz ve sonra kilitleyip anahtarını attığınız çelik bir kasa veya üzerine kalıcı kalemle yazı yazılmış bir taş gibi düşünebilirsiniz.

Örnek: Bir haftadaki gün sayısı. Üzerinde "Haftanın_Günleri" yazan kasanın içine 7 koyup kilitleriz. Program ne yaparsa yapsın, bir hafta asla 8 gün olamaz.



Değişken ve Sabitler

Robotik Sistemlerde Sabit Örnekleri

Robotun tasarımıyla veya evrensel kurallarla ilgili değişmemesi gereken veriler sabit olarak tanımlanır:

- **Maksimum_Hız:** Robotun güvenlik sebebiyle asla geçmemesi gereken en yüksek hız sınırı (Örn: 50 km/s).
- **Motor_Pini:** Robotun motorunun bağlandığı elektronik kart üzerindeki yuva numarası (Örn: 9 numaralı pin). Robot çalışırken kablounun yeri fiziksel olarak değişmeyeceği için bu numara sabittir.
- **Pi_Sayısı:** Dairesel tekerleklerin çevresini hesaplarken kullanılan 3.14 değeri.



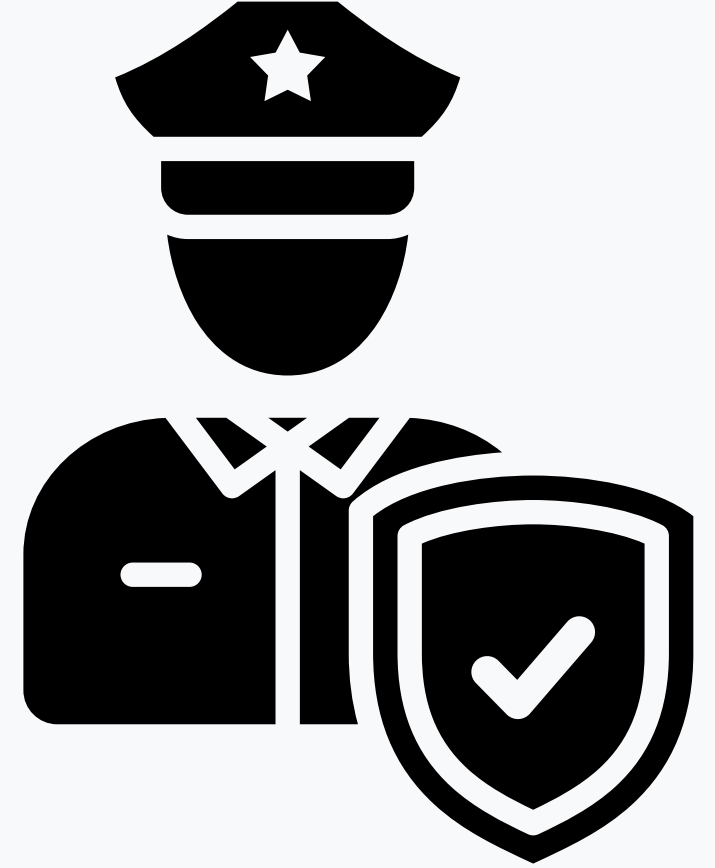
$$\pi = 3.14$$

Neden İkisine de İhtiyacımız Var?

Değişken mi, Sabit mi? Karşılaştırma

Güvenlik: Neden her şeyi değişken yapmıyoruz? Eğer robotun "Maksimum_Hız" sınırını değişken yapsaydık, bir hata (bug) veya dışarıdan gelen bir müdahale sonucunda bu değer aniden 500 km/s'ye çıkabilir ve robot parçalanabilirdi.

Kural: Sabitler, sistemin kurallarını korur. Değişkenler ise sistemin anlık durumunu yaşamasını sağlar.



DİKKAT!

Oyun tasarlarken ya da bir program yaparken şuna dikkat etmeliyiz:

- Eğer bir bilginin programın başından sonuna kadar aynı kalacağından eminseniz onu mutlaka **Sabit** yapın.
- Eğer o bilgi oyun oynandıkça, sensör ölçüm yaptıkça veya zaman geçtikçe artacak/azalacaksa onu **Değişken** yapın.



OKUL TEMELLİ PLANLAMA

Yerel Problem Çözümü

Okul Temelli Planlamada Ne Yapacağız?

“Yerel Problem Çözümü” etkinliği ile işleyeceğimiz okul temelli planlama etkinliğinde kendi okul veya mahalle çevrelerimizden seçebileceğimiz, algoritmik düşünceye uygun sorunlar bulacağız.

Bu sorunlar için algoritmik çözüm fikirleri düşüneceğiz.

Son olarak bulduğumuz çözümü algoritma olarak yazacağız.



ÖRNEK

Koridor Enerji Tasarrufu

Yerel Sorun: Teneffüsten sonra derse girildiğinde koridorlardaki ışıkların açık unutulması ve elektrik israfı.

Algoritmik Çözüm Fikri: Hareket ve ışık sensörlerine bağlı "Akıllı Koridor Aydınlatması".

Algoritma Adımları:

1. Başla.
2. Ortamdaki ışık miktarını ölç.
3. Eğer ortam aydınlıksa (gündüz ise) ışıkları kapalı tut ve Adım 2'ye dön.
4. Eğer ortam karanlıksa hareket sensörünü kontrol et.
5. Koridorda hareket varsa ışıkları yak.
6. Koridorda 5 dakika boyunca hiçbir hareket yoksa ışıkları kapat.
7. Başa dön.

