1) Thread ve process nedir ve aralarındaki farklar nelerdir, anlatınız.

Process (işlem)

İhtiyaç duyulan bütün kaynaklarla beraber belleğe yüklenmiş bir programdır :Register, Counter ve stack'tir. Her process 'in ayrı bir bellekte bulunmaktadır. Bunun anlamı her işlemin birbirinden izole olarak çalıştığıdır. Bu izole durumu çok önemlidir. Çünkü izole olması demek işlemdeki oluşmuş olan bir sorunun diğer işlemleri de bozmasını ya da zarar vermesini engellemektedir.

Register: Kayıt, saklama adresi gibi verileri tutabilir.

Counter: Bilgisayarın program esnasında nerede olduğunu inceler.

- Process çalışma durumunda hazır bekleyen programdır.
- Program bir komutlar bütünüdür. Process ise bu programın çalıştığı yaptığı işlemdir.

Processler arasındaki haberleşme :

- İş aktarımı(tünel/pipe)
- Adlandırılmış iş aktarımı (named pipe)
- Mesaj kuyruğu
- Semafor
- Paylaşım belleği
- Soket

Thread

Aynı process de birden çok işlemin yürütülmesine imkan tanımaktadır. Bir process çalışmaya başladığı anda direkt olarak bir thread oluşturulur. Bu Main Thread'dir.

Threadler iş parçacılarıdır. Processin alt kümesidir.

Birbirine bağımlıdır.

Ayrıca bir OS'de (İşletim Sistemi) gerçekleştirilebilecek en küçük işlem birimidir.

Çoklu görevlerde senkronizasyonu sağlamak için kullanılırlar.

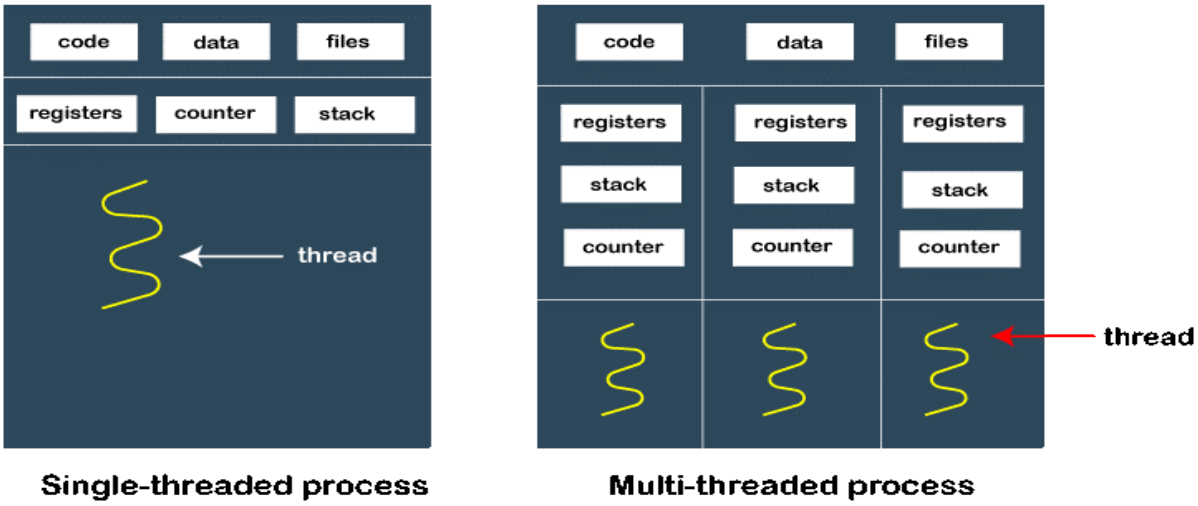
Bir process içerisinde birden fazla thread de bulunabilir. Buna Multi-Thread denir.

Parametre	Proses	Thread
Tanım	Çalışma durumunda hazır bekleyen programdır.	Prosesin bir parçasıdır.
Sona Erme	İş parçacığının sonlanması daha uzun sürer.	İş parçacığının sonlanması daha az zaman alır.
Oluşturulma Zamanı	Oluşturulması daha uzun sürer.	Oluşturulması daha az zaman alır.
İletişim	Prosesler arasındaki iletişim thread'e göre daha fazla zamana ihtiyaç duyar.	Threadler proseslere nazaran daha az zamana ihtiyaç duyarlar.
Bağlam Anahtarlama Süresi	Daha fazla zaman gerekir.	Daha az zaman gerekir.
Kaynak	Prosesler daha fazla kaynak tüketir.	Threadler daha az kaynak tüketir.
OS tarafında İşleyiş	Her farklı proses işletim sistemi tarafından ayrı ayrı işlem olarak ele alınır.	Tüm seviyelerdeki eşler işletim sistemi tarafından tek bir görev olarak ele alınır.
Hafıza	Prosesler çoğunlukla izoledir.	Threadler hafızayı paylaşırlar
Paylaşım	Verileri paylaşmazlar.	Threadler birbirleri ile veri paylaşırlar.

Threadin Avantaj Dezavantaj tablosu

Avantajlar	Dezavantajlar
Geliştirme süresinin kısalmasıyla gelişmiş performans sağlar.	Karmaşık hata ayıklama ve test işlemleri
Basitleştirilmiş ve aeorodinamik kodlama.	Bağlam anahtarlama durumunda ek yük.
Görevlerin eşzamanlı ver paralel şekilde yapılması.	Deadlock için potansiyel artış.
Kaynakları kullanarak önbellek depolamasını daha İyi şekilde kullanma.	Program yazarken artan zorluk seviyesi
Bakım maliyetinin az olması.	Öngörülemeyen sonuçlar.
CPU'yu daha iyi kullanma	

2) Multithread ve multiprocessing nedir ve aralarındaki farklar nelerdir, anlatınız.



Multiprocessing nedir?

Programın çalışması için birden fazla işlem çalışmaktadır. Bu işlemler farklı kaynaklar kullanmaktadır .Farklı işlemlerin kullanımdan kaynaklı programın işlem gücü artmaktadır. Farklı kaynakların kullanılmasıyla programın performansı arttırılmış olur.

- Process Control Blok(pcb) nedir, niçin kullanılır, neleri barındırır?
- Kendine ait bellekte yer ayırırlar. Birden çok iş parçacığından oluşabilir ve uygulamanın ta kendisidir.
- Process bellekte yer almasından dolayı birden fazla process olması maliyet doğurmaktadır
- Bilgisayarın işletim sisteminin herhangi bir process için tuttuğu değişik parametrelerdeki verilerin barındırdığı bir veri yapısıdır.

Yarış durumu: Sistemde proseslerin avantajlarından birisi olan bilgi paylaşımının bir noktada process senkronizasyon problemine dönüşmesi durumudur. Kısaca proseslerin aynı anda aynı veriyi aynı anda aynı veriyi değiştirmek istemesinden ortaya çıkan durumdur. Sorun çözümü senkronizasyondur.

Kritik bölge problemi: Sistemdeki proseslerin aynı anda kritik bölgelerinde işlem yapmaya çalışmasından ortaya çıkan sorundur.

Multithreading nedir?

Programın tek bir process üzerinden sağlanırken, bu process üzerinde birden fazla işlem parçacığı çalışmaktadır.

Process üzerinde çalışan bu çoklu işlem parçacıkları asenkron olarak farklı görevleri yerine getirmektedir için kullanılır. Bu sayede birbirlerini beklemesi gerekmeyen işlemlerin, tek bir process üzerinde asenkron olarak gerçekleştirilmeleri sağlanır ve performans arttırılmış olur.

Threading işlemlerini yöneten tipler System.Threading kütphanesindedir.

Start().Başlama

Sleep().threadin'i bir süre bekletir.

Threadler aynı anda aynı bölgeye girerse deadloclar meydana gelebilir.

Tek Thread kullanıldığı zaman bir işlem yaparken başka herhangi bir işlem yapılmamaktadır. O işlem bitene kadar başka kod çalışamaz.

Ama Multithreadin ile aynı anda birçok işlem yapılabilmektedir

Etkili bir şekilde kullanılmadığında programı yavaşlatabilir.

Thread Pool

Sistemde bulunan Threadleri kontrol altında tutabilmek ve yaratılmış olan Threadkein işi bittikten sonra Threadleri sonlandırmak yerine sonradan tekrar kullanabilmek için bekletmektir.

Farklar

- Multiprocess ile işlemler arası haberleşme konusunda farklı kaynaklar kullanıldığından ıpc kullanılmaktadır. Multithreadin is aynı sistem kaynağı kullanıldığı için işlem parçacıkları arası haberleşme daha kolaydır.
- **MultiProcessing ile birden fazla process oluşturularak işlem kapasitesi arttırılır.**
- MultiProcessing yapısında her bir process için ayrı ayrı bellek ve CPU zamanı ayrılır.

3. UDP nedir, nasıl çalışır; TCP nedir nasıl çalışır, aralarındaki farklar nelerdir anlatınız

Aktarım Denetleme Protokol TCP ve kullanıcı DATA PROTOKOLÜ UDP en popüler iki aktarım katmanı protokollerindendir. Paketlerin hata olmadan, kayıpsız ve düzgün şekilde hedefe iletilmesi görevini üstlenirler. İkisi arasındaki fark ise:

TCP'nin uygulara birden fazla çeşitte destek sağlarken, UDP'nin tek destek sağlamasıdır. Bu nedenle TCP, UDP'ye göre çok daha karmaşıktır.

TCP oldukça yavaş ancak fazlasıyla güvenli

UDP oldukça hızlı ancak yeterince güvenilir değil.

TCP, paketleri doğruluğunu ve güvenilirliğini sağlamak için onları parça parça ele alır ve kontrol eder

UDP de verilirin doğruşuğu garanti edilmese de ona nazaran daha hızlıdır.Video veya ses örneklerinde kullanılır.

TCP Üç Yönlü Bağlantı İsteği (iletişimi başlatmak için)

A sisteminin, B sistemiyle bağlantı kurmak istediğini varsayalım

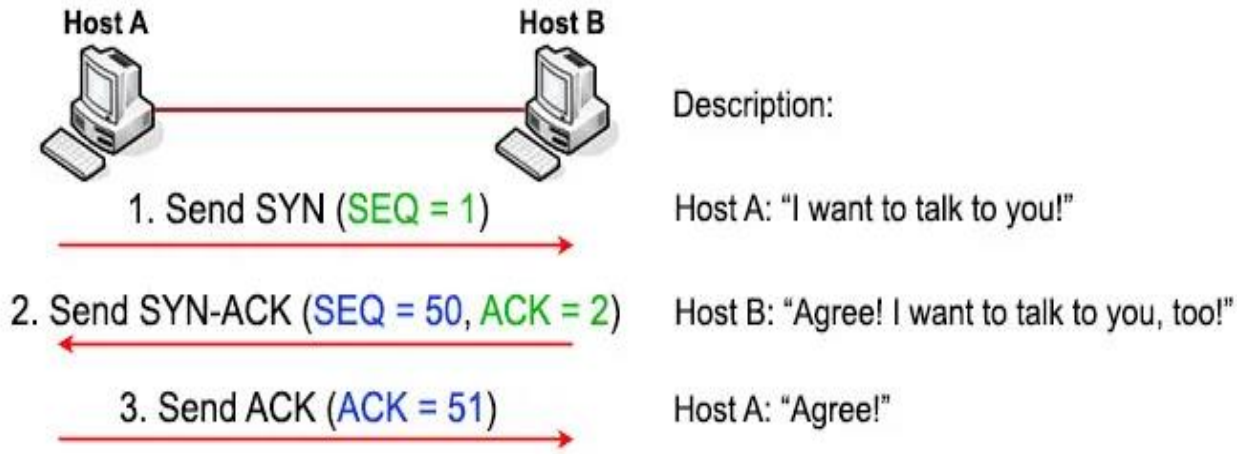
lk olarak A sistemi bir SYN mesajı gönderir (Senkronize nin kısaltılmış hali: SYN) . Bu SYN mesajı B sistemi ile bağlantı kurma isteğini içerir. Bu mesaj bir sıra numarası (SEQ) içerebilir. Bu sıra numarası herhangi bir 32 bitlik sayı olabilir (0'dan 232'ye) . Dolayısıyla bunu temsilen “X” kullanabiliriz.

2- A sisteminden SYN mesajı alındıktan sonra, B sistemi SYN-ACK mesajıyla bir cevap oluşturur (ACKnowledge = istek kabul). Bu mesaj bir SYN sıra numarası ve bir ACK numarasını kapsar:
*SYN sıra numarası (“y” ile gösterilir) rastgele bir sayı olabilir ve A sistemiyle bir ilgisi olmayacaktır.
* ACK numarası, A sistemindeki SYN numarasının bir değer üstündedir. Dolayısıyla bunu “x+1” şeklinde gösterebiliriz.

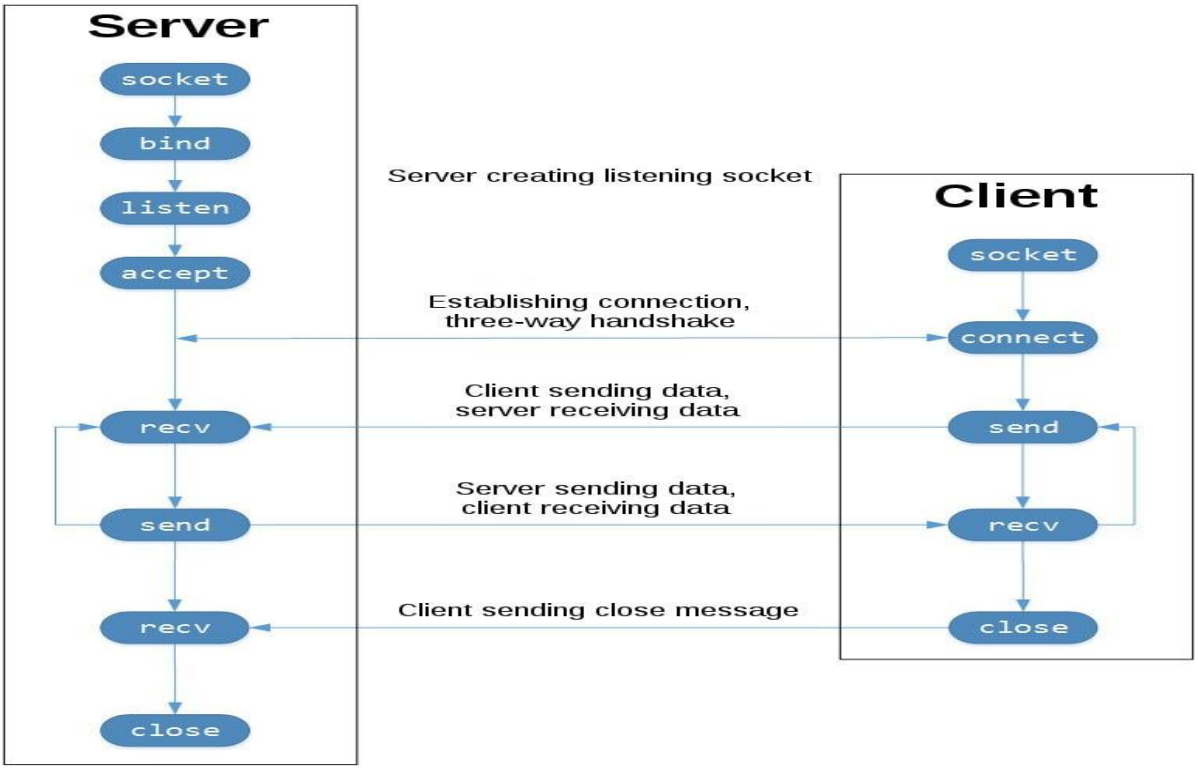
SYN-ACK mesajı, B sisteminin A sistemiyle iletişimi kabul ettiğini gösterir(ACK aracılığıyla). Ayrıca SYN aracılığıyla A sisteminin hala bağlantı kurmak isteyip istemediğini sorar.

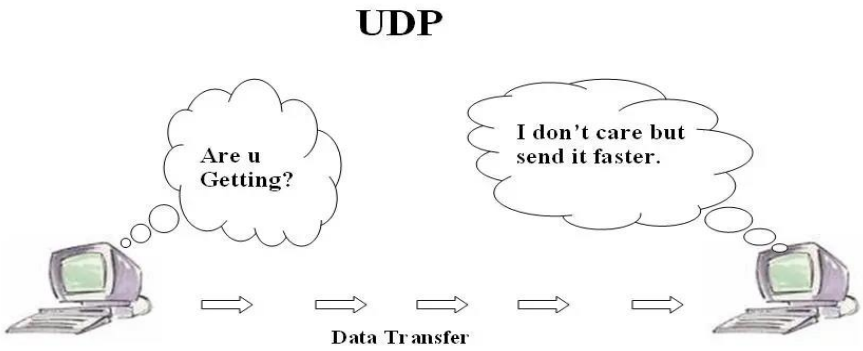
3- A sistemi, SYN-ACK mesajını aldıktan sonra geri iletim gerçekleştirir. SYN sıra numarasına ekleme yaparak hala bağlantı kurmak istediğini iletir (“y+1” ile gösterebiliriz).

Grafiksel olarak göstermek gerekirse (x=1 , y=50):



TCP

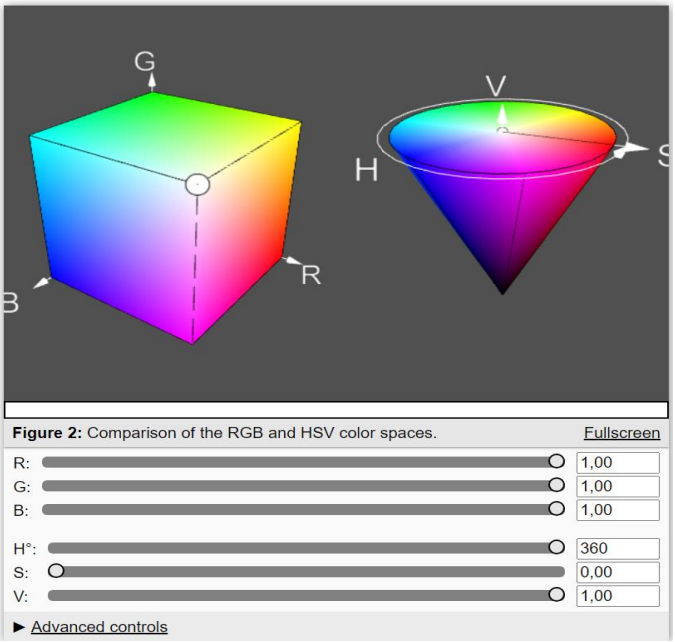




4) OpenCV'de renk filtreleme ve maskeleme işlemleri nasıl yapılır araştırınız.

Renk filtreleme işleminde alınan görüntüyü HSV (hue[ton], saturation[doygunluk], value[değer]) renk modeline çevirmemiz gerekmektedir.

Genel HSV renk alanı, RGB renk uzayının başka bir ifadesidir. Algılama renk kontaklarını RGB'den daha net ifade ettiğinden ve hesaplama çok basit olduğundan görüntü işlemede yaygın olarak kullanılmaktadır. Bu sebepten ötürü HSV renk modelini kullanacağız.



Öncelikle aldığımız görüntüyü HSV'te çevirelim.

```
hsv=cv2.cvtColor(bilgisayarKamerasi,cv2.COLOR_BGR2HSV)
```

Mavi rengi için alt ve üst referans değerlerini RGB renk kodları ile belirledik. Bu renk kodlarının aralıklarındaki değerlere göre filtreleme işlemini gerçekleştireceğiz. Mavi rengin referans değerlerini belirledikten sonra görüntü üzerinde maskeleme yapacağız.

Maskeleme, tanımladığımız küçük görüntü parçası ile daha büyük bir görüntüyü değiştirmek için kullandığımız bir görüntü işleme yöntemidir. Maskeleme, kenar algılama, hareket algılama ve gürültü azaltma dahil olmak üzere birçok görüntü işleme türünün altında yatan bir işlemdir.

```
baslangic_altMavi=np.array([100,60,60])

    bitis_ustMavi=np.array([140,255,255])

    maske=cv2.inRange(hsv,baslangic_altMavi,bitis_ustMavi)
```

5) Ekte gönderilen "16.jpg" ve "26.jpg" isimli görüntülere gerekli renk filtreleme işlemlerini yaparak yalnızca rakamların olduğu görüntüleri elde eden programı yazınız.

6) Ekte gönderilen "Referans" klasörü içerisindeki programın nasıl çalıştığını inceleyiniz. Linkteki rehber videodaki gerekliliklere göre programın nasıl çalıştığını anlatan bir rapor oluşturunuz.

SERVER
Server oluşturup client da mesajları alırız
import queue: Çoklu işlem ve çoklu iş parçacığı programlamasında yaygın olarak kullanılır
*queue.Queue: Tipik kuyruk yapısını uygular. itemler kuyruğun sonuna eklenir ve kuyruğun başından alınır.
*Queue.LifeQueue: Bu ise son giren ilk çıkar.
Self.server = UDP olarak socket'a geçirilir
Self.server.bind = server oluşturlur

Def reviece:
#recv işlevi sadece veriyi alırken, recvfrom işlevi ayrıca bu verinin geldiği kaynağı da döndürür.

#message: Bu değişken, soketten alınan veriyi içerir. recvfrom işlevi, maksimum 1024 byte uzunluğundaki bir veriyi message değişkenine atar

#addr: Bu değişken, gelen verinin kaynağını temsil eder. İki öğeli bir tuple (demet) olarak döner. İlk öğe, kaynak IP adresini içerir ve ikinci öğe, kaynak port numarasını içerir. Bu bilgiler, gelen verinin nereden geldiğini belirlemenize yardımcı olur.

Try except ile adres ve mesaj alınır
Def broadcast:
Receiveden alınan mesaj boş olmadığı sürece mesajlar alınır ve decode ile döndürülür

CLİENT
portSend ve port vardır port clientın PortSend ise send işlevi için gerekli olan porttur.
Self.client ile UDP client oluşturulur.
Bind ile client bağlanır.
Send ile encode edilmiş mesajı localHostte portsend ile gönderilir gönderir.

GENERATE
İimg adında 900,900 lük pencere beyaz pencere oluşturulur
Başta neden xve y Corp oluşturulmuş anlamadım
While ile sürekli olacak şekilde pencere içerisine daire oluşturulup yenilenip kay circle.pngye kaydedilir.

CONTROLSTATİON
sv ile Servera status ile clienta giriş yapılır.
dataThread1:serverdan mesajı alır dataThread2:serverdan mesajı verir
statusThread: durumu clientdan gönderir

```
def start(self):
```
self.flag= 0'lanır
clienttaki 0 olan self.massageyi "off" yapar

```
def getLoc(self):
```
Tkineter oluşturulur
Başlatbutonu getLoca gider:"off" olan self.massageyi "on" yapar
Serverın msg si cismin yönünü bulur(Camera tarafından)
Flag sayesinde fonksiyon Sürekli olarak getLoc fonksiyonu tekrarlanır

Bitir butonu ile stopGetLoc gidilir

```
def stopGetLoc(self):
```
flag 1 olur ve getLoc durur

CAMERA
Client ile clienta
Status ile servera girer

DataThread: iş parçacığı client.send e girer ve start başlatır
statusThread1:serverdan veri al statusThread2:serverdan veri ver ve start la başlatılırlar

```
def find(self):
while True ile fonksiyon tekrarlanır
```
ControlStationla “on” olan msg ile fonksiyon başlar
İmg ile generate sürekli oluşturulan resimler okunur
Cisimin yeri tespit edilir dışına kare yerleştirilir ve dairenin merkezine nokta yerleştirilir.
getLoc fonksiyonu ile alınacak bilgi dairenin merkezine yerleştirilir.
Ve img gösterilir ama öncesinde getLocun gönderdiği mesaj:

```
def getLoc(self,origin, target):
```
merkezi target olarak gönderdik ve orijine de pencerenin merkezini
Bunların birbirlerine göre durumları karşılaştırılır ve o değeri döndürür.

7)
Landing.py adında txt yazdırma işlemini ayrıyeten bir py dosyasında yaptırmaya çalıştım ancak servera bağlanma hatası verdi bende ladingcamera üzerinden bu işlemi yaptırdım. Burda da yazım hızını yavaşlatmaya çalıştım ancak time.sleep(5) fonksiyonu işe yaramadı.

https://thecodeprogram.com/multithreading-ve-multiprocessing-nedir--farklari-nelerdir-#:~:text=MultiThreading%20ile%20tek%20bir%20process%20%C3%BCzerinde%20birden%20fazla%20i%C5%9Flem%20par%C3%A7ac%C4%B1%C4%9F%C4%B1,i%C3%A7in%20daha%20az%20kaynak%20t%C3%BCketir.

https://blog.burakkutbay.com/thread-ve-process-arasindaki-farklar.html/

https://www.ozztech.net/yazilim-gelistirme/thread-ve-process/

https://medium.com/@keparlak/isletim-sisteminde-process-ve-thread-27bfdb9810f2

https://realpython.com/python-sockets/

https://www.mobilhanem.com/opencvde-filtreleme-islemlerine-giris/

https://www.mobilhanem.com/opencvde-filtreleme-islemlerine-giris/

https://teknogof.com/nedir/udp

https://medium.com/unknown-source/tcp-udp-giri%C5%9F-e%C4%9Fitimi-4aeb4cd558e5