1. **VS CODE VE GIT-SCM KURULUMU**

* Visual Studio Code ve GIT-SCM Versiyon Kontrol Sisteminin kurulumu tamamlandı.
* Git komut ekranında hem Linux komutları hem de Windows komutları çalışır. (ls ve dır gibi)
* Komut ekranından masaüstünde bir klasör oluşturup, o klasörü VS Code içerisinde açmak için aşağıdaki komut satırlarını kullanmamız gereklidir.

cd Desktop

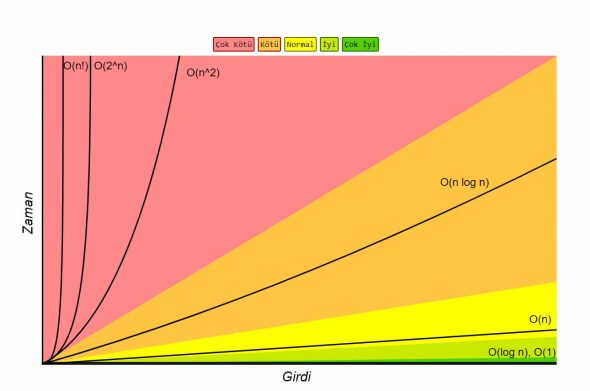
mkdir DOTNETCORE

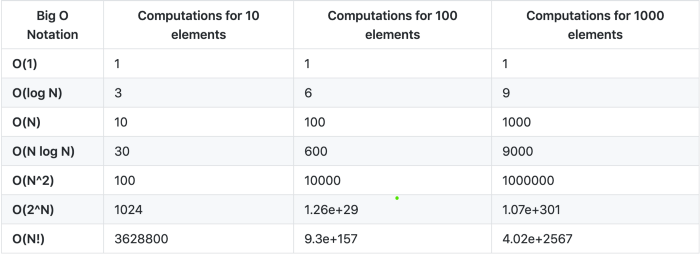
cd DOTNETCORE

code .(bu komut çalıştığında VS Code açılır.)

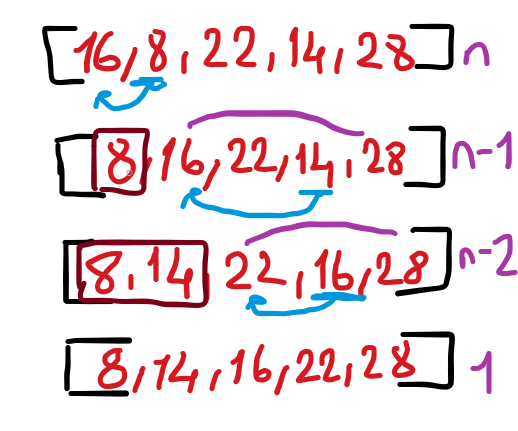
* VS Code içerisinde sol tarafta bulunan kaynat denetimi kısmından projemizi direkt olarak githuba yükleyebiliriz.
* VS Code’a özel iki özellik vardır. Bunlardan bir tanesi; IntelliSense, yazdığınız kodu kolaylıkla tamamlayabilmenizi ve eğer bir fonksiyon kullanıyorsanız o fonksiyonun özelliklerini anlayabilmenizi sağlıyor. Diğeri ise; Emmet, içerisinde bulundurduğu kısaltmalarla birlikte, kod yazarken zaman kazanmamızı sağlıyor.
* VS Code içerisinde birden çok satırı yorum satırı haline getirmek için shift+ alt+ A yapmamız gereklidir.

1. **VERİ YAPILARI VE ALGORİTMALAR**

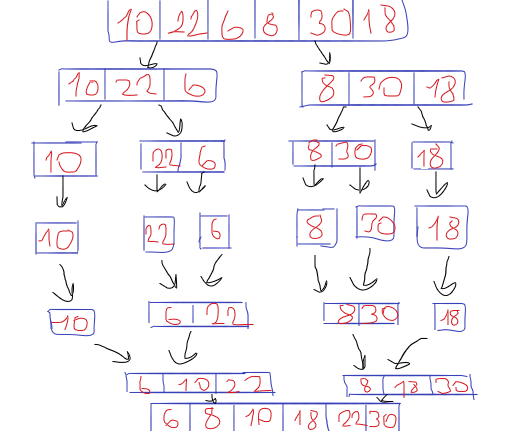
* Algoritma; bir problemi çözmek için gerekli olan adımların bütününe verilen addır.
* Bilgisayar bilgiyi (Resim, ses, yazı vb) ifade etmek ve döngüyü sağlamak için bit (0 ve 1)' den oluşan ikili sayıları (Binary Numbers) kullanıyor.
* İkili sayılarda bulunan 1 ve 0 rakamları (bit), bilgisayarın elektrik iletimi için kullandığı transistörlerin açık veya kapalı olma durumunu gösteriyor. Transistörlerde iki tane komut vardır, 0 (kapat) ve 1 (aç).
* Bilgisayarda sayısal olmayan veriler de 0 ve 1 şeklinde tutulur.
* Bilgisayarda tutulan verilerin en küçük yapı taşı bitlerdir.
* 1 Byte = 8 bit
* Binary semboller kişiden kişiye göre farklılık gösterirler. Mesela 00 bana göre k’yı sembolize ederken sana göre m’yi sembolize edebilir.
* Array; birden fazla veriyi bir veri tipinde saklamamızı sağlayan, belirli bir uzunluğa sahip olan veri yapısıdır.
* Dynamic Array; belirli bir boyuta sahip olamayan dizi demektir. Dezavantajı hafızada çok yer kaplar.
* Linked List; yan yana olma zorunluluğu olmadan verileri hafızada tutmaya yarayan veri yapısıdır. Yeni gelen eleman için hafıza yer açılmaz. Array’den farklı olarak veriler hafızada dağılmış şekilde tutulur. Fakat son gelen eleman bir önceki elemana adresini bildirmek zorundadır. Yani her bir düğüm bir sonrakinin adresini tutar.
* Linked List’teki düğümler veri ve adres tutar.
* Arrayler hafıza sadece veri tuttuğu için daha az yer kaplar Linked List’e göre.
* Ekleme çıkarmanın fazla olduğu durumlarda Linked List tercih edilir.
* Stack, LIFO (Last in First out) (En son giren en önce çıkar) mantığına dayanan, elemanlar topluluğundan oluşan bir yapıdır. Push, yığının üzerine eleman eklemek için kullanılır (Koliye kitap koymak). Pop ise, yığından eleman çıkarmak için kullanılır.
* Queue (Kuyruk), FIFO (First in First out) (İlk giren ilk çıkar) prensibine dayanan, girişlerde ve çıkışlarda belirli bir kurala göre çalışan yapıdır. Queue (Kuyruk)'da eleman eklemesi yaparken enqueue methodunu kullanıyoruz. Eleman silerken ise dequeue methodunu kullanıyoruz.
* Algoritma analizi, var olan kaynaklara göre en uygun algoritmayı seçmek için uygulanır. Algoritma analizi, var olan kaynaklara ve programlama dillerine bağlıdır.
* Bir algoritmanın analizini yapmak için tek bir tane ortam kuruldu, çünkü algoritmalar başka cihazlarda farklı performans gösterebilir. Bunu engellemek için, bir algoritmanın gerçek performansını bulmak için RAM modeli geliştirildi. Ram, algoritmalar arasındaki farkları belirlemek için kullanacağımız bir araç olacak. Her işlemin birim zamanı var. Döngüler, kaç defa işlem yapıyorsa, (işlem sayısı \* kaç kere tekrar edeceği) kadar birim zaman alır. Toplama, Çıkarma, and, or gibi aritmetik işlemler, 1 birim zaman alır.
* Algoritma analizi 3 ana başlık altında incelenir; worst case, average case ve best case. Algoritmanın çalışmasını en iyi yansıtan average case’dir.
* Time Complexity; bir algoritmanın çalışması için gerekli olan süredir.
* Big O-Notation; yazılan bir algoritmanın performansını ölçmemiz için kullanılan en önemli araçlardan biridir. Big-O notation bir algoritmanın performansını veya time complexity’sini hesaplamak için kullanılır.
* **O(1)=** Constant complexity’de elinizde bulunan veri seti ne kadar büyük olursa olsun, çalıştırılma süresi ve kullanılan kaynak her zaman sabittir.
* **O(N)=** Linear complexity’de, elimizdeki veri seti arttıkça, çalıştırılma süreside doğru orantılı şekilde artış gösterir.
* **O(N^2)=** Quadratic Complexity input büyüklüğünün karesi ile doğru orantılıdır.
* **O(logN)=** Logarithmic time complexity, genelde her seferinde problemi ikiye bölen algoritmalarda kullanılır.
* **O(c^N)=** Base değeri (c) 2 olan Exponential Time Complexity’de, input N büyüklüğünde ise output 2^N olur.

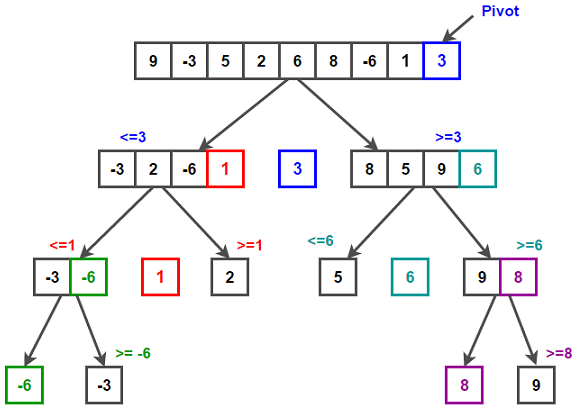
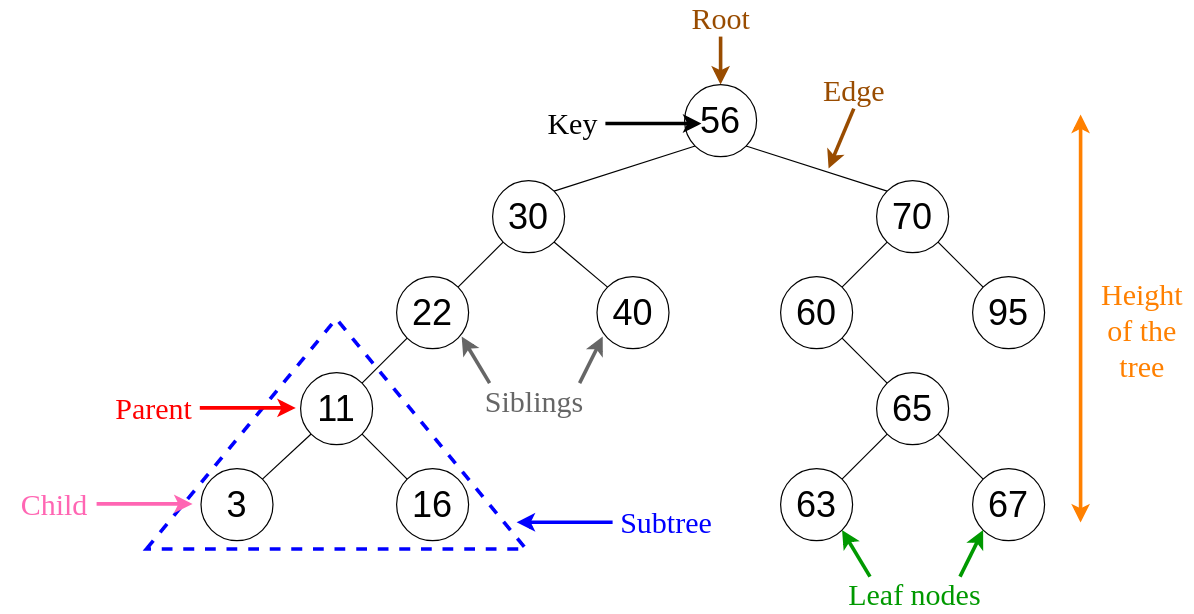


* **Sorting;** bir eleman dizisini belli sıralama kurallarına göre sıralar.
* **Insertion Sort;** en basit sıralama algoritmasıdır. Verilen dizideki en küçük elemanı buluyor ve ilk sıradakiyle yer değiştiriyor, bunu dizi sıralanana kadar yapıyor. Time complexity n^2 dir.



* **Merge Sort;** Bir listeyi her adımda parçaya ayırıp tek eleman kalıncaya kadar bölüyor. Böldükten sonra sıralı bir şekilde bize sunuyor. Time complexity nlogn dir.



* **Quick Sort;** İlk olarak bir pivot belirler bu pivota göre pivottan küçük ve eşitler sol kısmına, pivottan büyük ve eşitler sağ kısmına yazılır. Parçalanmış kısımlar yeni bir pivot belirlenerek parça pinçik edilir. Hızlı sıralama günümüzde çok yaygın olarak kullanılan bir sıralama algoritmasıdır. N tane sayıyı average case e göre big-o nlogn, worst case e göre big-o n^2 karmaşıklığı ile sıralanır.
* **Searching;** Arama algoritmaları istediğim özellikteki verinin elimdeki veri setlerinde aranıp, bulunup getirilmesi demek. Bunun hızlı olmasına önem gösterilir.
* **Linear Search;** tek tek elemanları dolandıktan sonra istediğim elemanın olup olmadığına bakmaktır. Worst case durumunda time complexity O(n) dir.
* **Binary Search;** elimizde bulunan veri dizisini sıralı olduğunu varsayıyor, diziyi her seferinde ikiye bölerek ikili arama yapar. Sıralı bir listem var ise benim Big-o logn olarak karşımıza çıkıyor.
* **Binary Search Tree;** Ağacın Sağ tarafından kendinden büyük elemanlar, sol tarafında ise kendinden küçük elemanlar bulunacak.
* Tree'ye eleman eklemek istediğimde root'dan başlıyorum. Örnek olarak ben 26 sayısını ağaç yapısına eklemek istiyorum. Root'a soruyorum senin değerin ne 56. Sağ tarafında kendinden büyük, sol tarafında kendinden küçük elemanlar var. O yüzden sırasıyla 56 ve 30 a kadar ilerliyorum. 30 bana benim sol tarafıma geçmelisin çünkü sen benden küçüksün diyor. Karşıma 22 değerinde olan düğüm çıkıyor ve 22 den büyük olduğum için sağ tarafına bir köşe çekiyorum ve 26 sayısını bağlıyorum.

1. **C#**

* Orta seviyeli bir programlama dili olan C# Microsoft tarafından geliştirilmiştir. Günlük kullanım diline de makine diline aynı yakınlıktadır. İlk Alfa versiyonu 2000'li yıllarda ortaya çıkmıştır. Java ve C++ ın güçlü yönlerinden esinlenmiştir. Tasarlanırken kullanım amacına uygun olması ve kolay adapte olunması göz önünde bulundurulmuştur. Nesne yönelimdir bir dildir. Dolayısıyla C++ veya C kullanan insanlar kolayca adapte olabilmektedir.
* .NET veya .NET Framework Microsoft firması tarafından 2000 yılında geliştirilmeye başlanmış ve .NET Foundation topluluğuna devir edilmiş bir yazılım geliştirme platformudur. İçerisinde yer alan kütüphaneler sayesinde kolayca masaüstü, web ve cep telefonu uygulamaları geliştirmeye imkan verir.
* .NET 5 kurmak için bir sdk dosyası indirip kurdum.(VS Code içerisinde C# kullanmak için bazı extension’ların kurulması gerekir. (Net Core Tools (Extensison Id : formulahendry.dotnet), C# (Extension Id : ms-dotnettools.csharp), Ms Build Project Tools (Extension Id : tintoy.msbuild-project-tools))
* VS Code’da .NET projesi oluşturmak için komut ekranını kullanacağız. Bunun için teker teker şu komutları yazmalıyız. (1- cd Desktop/DOTNETCORE/KonsolProgramlama 2- dotnet new console) Böylece oluşturduğumuz klasörün içerisine konsol uygulamasını otomatik olarak oluşturdu.
* VS Code’un içerisine oluşturulan bu klasörü taşıyalım ve C# kodlamaya başlayalım.
* Main metot: bir programın başladığı ve bittiği yer.
* Using System kütüphanesi Console sınıfını içerisinde barındırır. Eğer ben o kütüphaneyi yazmayı unutursam en başa Console.writeline çalışmaz hata verir.
* Yazd