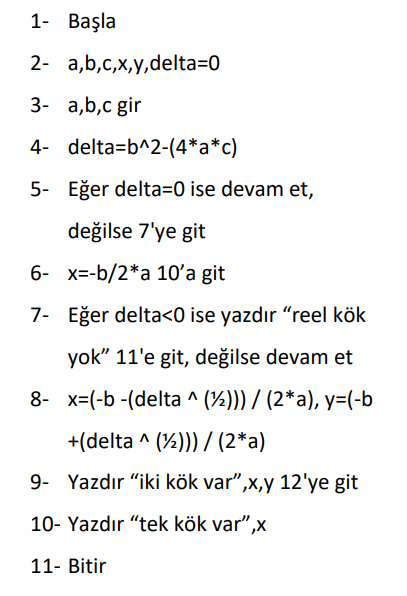
**SORU-)** ax2 + bx +c = 0 tipindeki bir denklemin köklerini bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** ax^2+bx+c tipindeki bir denklem için a, b, c değerlerini dışardan girerek Delta değerimizi hesaplarız. Delta değerimizin 0 veya 0’dan küçük ya da büyük olma durumuna göre de denklemin köklerini buluruz. Değerleri alıp formüle uygularız. Direkt matematik ve iki bilinmeyenli bir denklem formülüne değerlerimizi uygularız. Bu işlemi sadece Delta’yı karşılaştırmak için kullanırız.

**CEVAP-)**

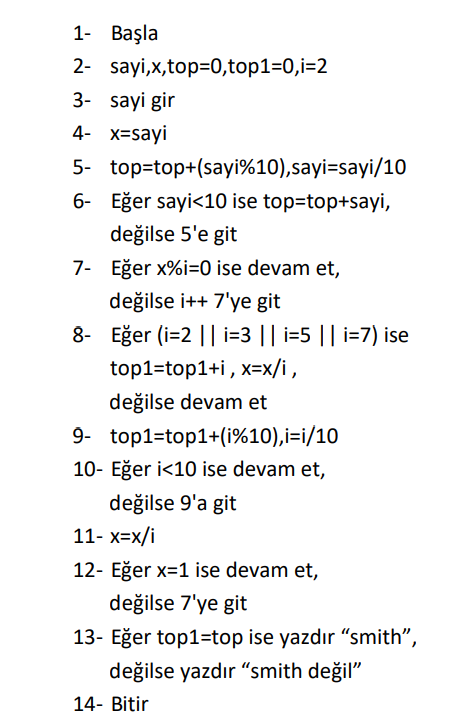
****

**SORU-)** Girilen sayının smith sayısı olup olmadığını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

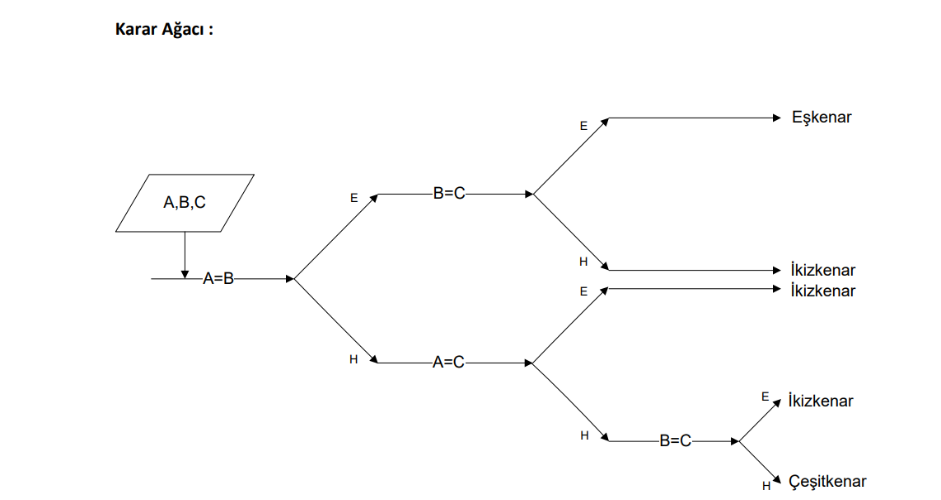
**Açıklama:** 1 den büyük asal olmayan bir tam sayının rakamlarının toplamı, sayı, asal çarpanlarına ayrılarak yazıldığında bu yazılışta bulunan tüm asal çarpanların rakamlarının toplamına eşit oluyorsa bu tür sayılara Smith sayısı denir. 121 = 11 \* 11 1 + 2 + 1 = 1 + 1 + 1 + 1 4 = 4 (121 bir Smith sayısıdır.) 166 = 2 \* 83 1 + 6 + 6 = 2 + 8 + 3 13 = 13 (166 bir Smith sayısıdır.) Bu sayının nasıl ortaya çıktığını merak ediyor musunuz? 1982 yılında matematikçi Albert Wilansky, kardeşi Smith’i ararken onun telefon numarasının (4937775) bu ilginç özelliğini fark etmiş. Bundan dolayı da bu sayılara Smith sayıları adını vermiş. Bu sayıyı da inceleyelim; 4937775 = 3 \* 5 \* 5\* 65837 4 + 9 + 3 + 7 + 7 + 7 + 5 = 3 + 5 + 5+ 6 + 5 + 8 + 3 + 7 42 = 42 (4937775 bir Smith sayısıdır.)

**CEVAP-)**

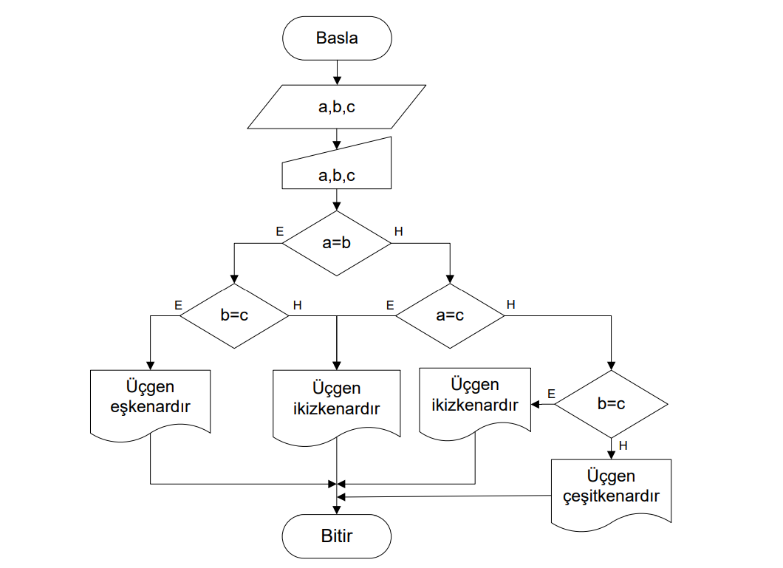
****

**SORU-)** İç açıları verilen üçgenin karar ağacı, algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Üçgenin karar ağacı yapısı ile ilgili bu soru, karar (eğer) yapılarını iyi öğreten bir soru tarzıdır. Bir üçgenin 3 kenarı vardır. Bu kenarların eşitliği veya eşitsizliği durumunda üçgen ya eşkenar ya ikizkenar ya da çeşitkenardır. Yukarıdaki şekilde de bunu açıkça görebiliriz. Soruda 3 kenarı dışarıdan giriyoruz. Eşit olma ya da olmama durumuna göre üçgenin tipini ekrana yazdırıyoruz. 

**Cevap**

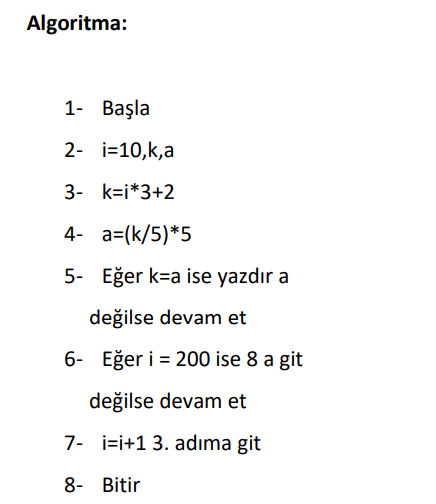


**SORU-)** 10 ile 200 arasındaki tam sayılardan 3 katının 2 fazlası 5 ile tam bölünebilen sayıları gösteren programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bu soruda bir döngü olduğu hemen aklımıza gelmelidir. Çünkü bir sayı aralığı verilmiştir. (10 ile 200 arasında gibi.) Akış diyagramını çizerken döngü şeklini kullanmadık fakat ileri ki sorularda döngü şekli ile işlemimizi daha kolay yapabilceğiz. Bu, algoritma açısından da kolay olacaktır, sonuç da günlük dil kullanmaktayız. Bu tarz sorular aslında sizleri problem çözmeye alıştırmak için yapılmış basit sorulardır. Bu soruda da döngü vardır ve sayacımız yine (i)’dir. Bu i’yi unutmamak gerekir. Sayımız 10 için geldiğinde elimizde k=10\*3+2= 32

**CEVAP-)**

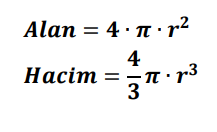
****

**SORU-)** Dışarıdan yarıçapı girilen kürenin alanını ve hacmini hesaplayan programın algoritmasını yazınız.

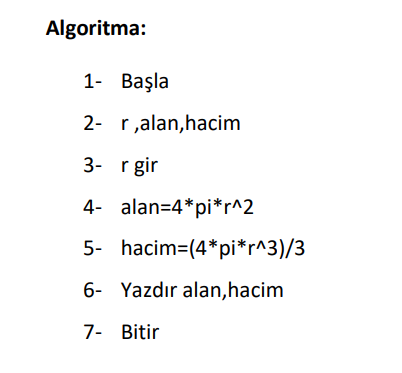
**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Koşul

**Açıklama:** Bir önceki soruda olduğu gibi girilen değerleri formülde kullanıp sonucu ekrana yazdırıyoruz. Yarıçap değeri dışarıdan girilecektir, zaten pi sayının değeri sabit (consst) 3,14 olduğundan bu değerler ile kürenin alan ve hacmini buluyoruz.

**FORMÜL-)**



**CEVAP-)**

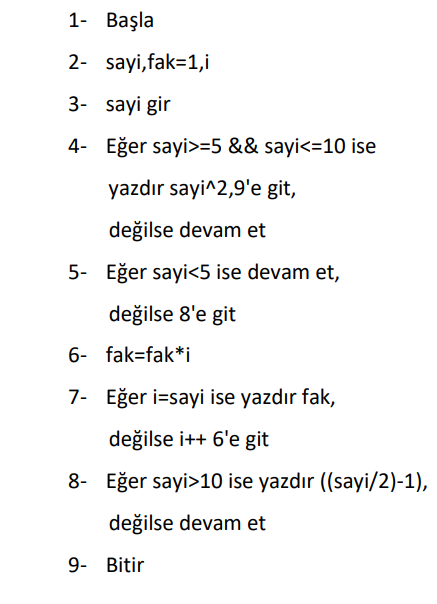
****

**SORU-)** Ekrandan girilen bir sayı eğer 5-10 arasında ise girilen sayının karesini alıp gösteren, eğer 5'ten küçük ise faktöriyelini alan, 10'dan büyük ise sayıyı ikiye bölüp bir eksiğini yazan programın algoritması

**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bu soruda if (eğer) önemli rol taşımaktadır. Çünkü 3 adet seçeneğimiz bulunmaktadır. Bunlardan birini seçmek için eğer kullanmalıyız. Faktoriyel, döngü ister ama diğer kare alma (^) ve sayıyı ikiye bölüp bir eksiğini alma işlemi, direkt işlemi yapıp ekrana basmakla sona erer. Seçimlik sorular ileride işimize yarayabilir. Mesela şu işlem için 1’e bas, şu işlem için 2’ye bas gibi program menu girişlerinde if yada case mekanizmalarını kullanırız. Bunun için önemlidir.

**CEVAP-)**

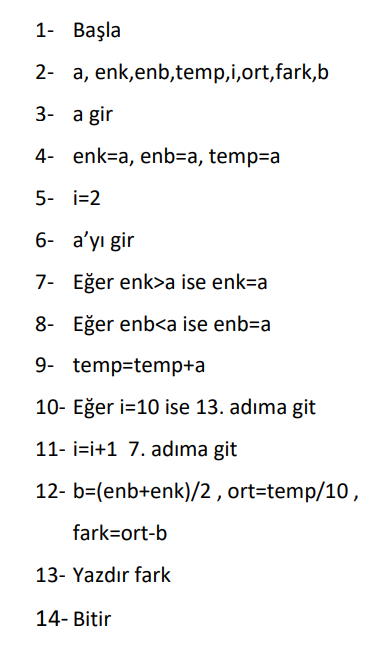
****

**SORU-)** Arka arkaya girilen rastgele 10 tam sayının ortalaması ile bu sayılardan en büyük ve en küçük olanın ortalamasını bularak elde edilen bu iki ortalamanın farkını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Soruda arka arkaya girilen 10 sayı dediğine göre bir sayaç değerimiz olmalıdır. Bunun için de her zaman belirttiğimiz gibi I kullanırız. I=1 den 10 oluncaya kadar bir döngü oluşturur. Bunları top denilen bir değişkene her defasında bir önceki ile toplaya atarız ve 10 bölüp ortalamayı buluruz. Döngüyü kullandık. Şimdi bir de takas algoritmamızı kullanarak soruyu çözelim. En büyük ve en küçük gibi sorularda unutulmamalıdır ki enb veya enk diye bir değişken alırız, ilk sayıyı bunun içine atarız ve diğer kaç sayı gelecekse enb ile karşılaştırırız. Bu değerden büyük enb içindeki değer değişir. Enk için de bu geçerlidir. Bunu da yaptıktan sonra enb ve enk toplayıp 2’ye bölüp ortalamayı buluruz. İlk ortalama ve son ortalama değeri çıkarıp sonucu ekrana yazdırırız. Burada fark da eğer ‘e sokulsaydı tüm mekanizmaları barından bir soru elde edilebilirdi. Fakat sonuçta eğerin (karar yapılarının) öğrenildiğini ümit ediyoruz.

**CEVAP-)**

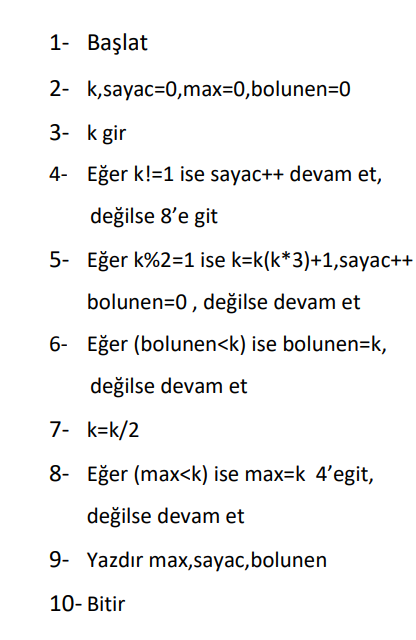
****

**SORU-)**1 k sayısı tek ise 3 ile çarpılıp 1 ekleniyor çift ise 2 ile bölünüyor işlem k sayısı 1 olana kadar devam ediyor bu işlemin kaç adım sürdüğünü, işlem, işlem sırasında k sayısının aldığı max değeri k sayısının hangi sayıdan sonra hep çift olarak 1'e ulaştığını bulan programın algoritmasını yazınız.

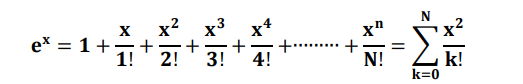
**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bu soru ilk sorduğumda öğrencilerimin kafasını çok karıştırmıştı ve ancak birkaçı cevap verebilmişti. Aslında sorudaki işlemler gayet basittir ama çözüm yeteneği gelişmiş kişilerin yapabileceği türden bir sorudur. Çünkü algoritma ya da problem çözme biraz da çalışma azmi ve yetenek ister kanaatindeyiz. Bu soru sayılarla oyun yapılan ilgi çekici bir sorudur. Bir K sayısı giriliyor ve bu k sayısı 1 olana kadar bazı işlemler uygulanıyor. Bu sorunun kodunu yaparken While kullanmak bizce en uygunu olacaktır. Direkt algoritma testine geçiyoruz. Bu algoritma testleri de sizlere programın nasıl çalıştığını gösterir. Adımlarla karşılaştırarak algoritmayı daha iyi anlayabiliriz.

**CEVAP-)**

****

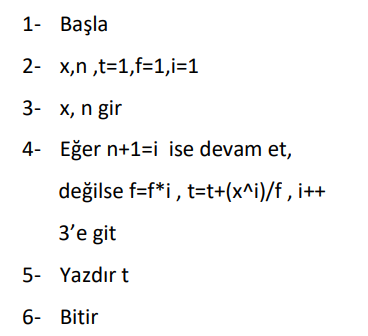
**SORU-)** e^x fonksiyonunun seriye açılımı aşağıdadır. Buna göre; dışarıdan girilen x ve N değerine göre; e i x ' hesaplayan programın algoritmasını yazınız.

****

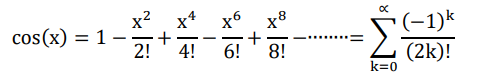
**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü

**Açıklama:** Bu soru seri örneklerimizden, içinde faktoriyel işlemini de barındırmaktadır. Çünkü faktoriyel yine algoritma sorularında döngü mekanizmasını anlamak için önemlidir. Burada fak=fak\* gibi kuralları her zaman akılda tutmak gerektiğinin önemini bir kez daha vurguluyoruz. Bu soru da verilen N ve X değerine göre e^x işleminin sonucunu veren bir algoritmadır. Bunu iki şekilde çözdük. Çünkü önceki sorularda algoritma yazarken döngü kullanmamıştık ama konu üzerinde kendimizi geliştirdikçe sorularımızda döngü kullanmamaya çalışmalıyız. Zaten unutulmamalıdırki algoritma çözüme giden adımları oluşturur. Bu soruda iki tane değişken tanımladık: Bunlardan birini faktoriyel için birisini de işlemin sonucunu tutan T değişkeni olarak ayarladık ve 1 değerinden başlattık. İşlem N kadar dönecek, her N değerindeki sonucu T değişkeninin içine atacak, sonra T yi ekran basacak ve işlem tamamlanacaktır.

**CEVAP-)**

****

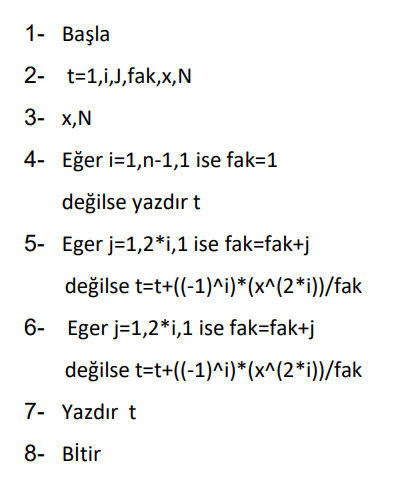
**SORU-)** Cos(x) fonksiyonu seriye aşağıdaki gibi açılmaktadır. Buna göre dışarıdan girilen x değerinin cosinüs’ünü hesaplayan programın algoritmasını yazınız.

****

**KULLANICAKLARINIZ-)** Interface (Arayüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bilgisayarın temeli matematiktir fakat formülü verilmiş bir işlemi programlamak basittir. Çünkü dışarıdan N ve X değerini alacağız ve bu seriyi bu iki değere göre hesaplayacağız. Daha önce de böyle sorular yapmıştık. Bu soruları sıkça ele almamızın amacı alıştırmaları bol bol yaparak programalama konusunda giriş kısmından orta seviyelere geçilebilmesidir. Bu sorunun algoritmasında olmasa bile akış diyagramında döngümüzü kullanabiliriz. Zaten program kodlarına bakıldığında hep döngü komutları görülecektir.

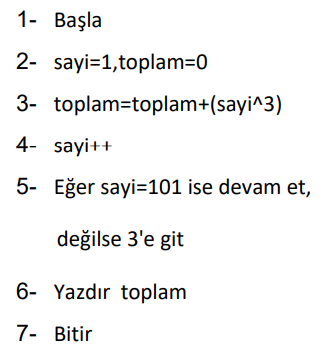
**CEVAP-)**

****

**SORU-)** 1'den 100'e kadar olan sayılarının küplerinin toplamını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Inheritance (Kalıtım), Döngü Koşul

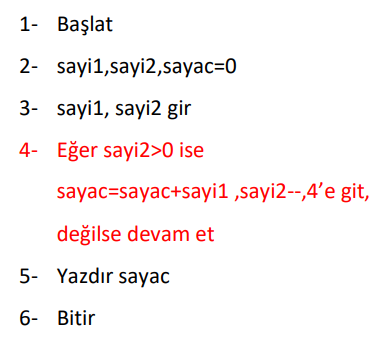
**CEVAP-)**

****

**SORU-)** Çarpma işlemini toplama kullanarak bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz) Koşul

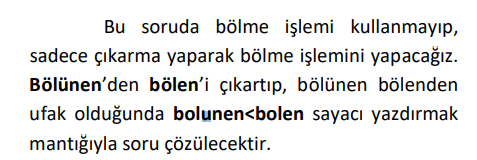
**CEVAP-)**

****

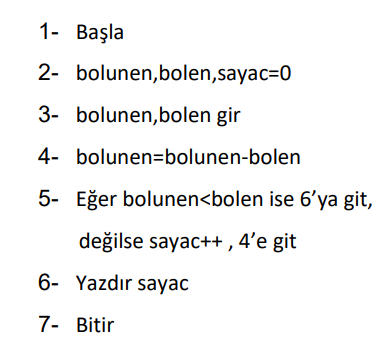
**SORU-)** Bölme işlemini çıkarma kullanarak yapan programın algoritma ve akış diyagramını çiziniz.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz) Koşul

**Açıklama**:

****

**CEVAP-)**

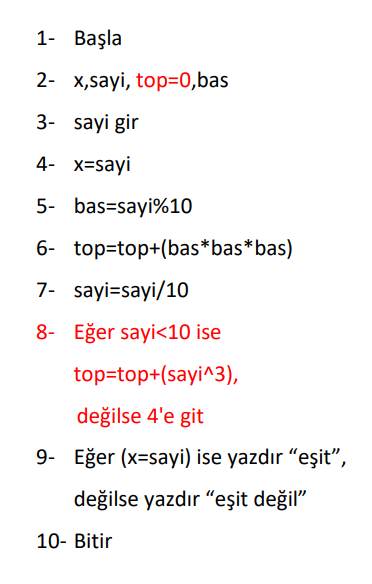
****

**SORU-)** Girilen 3 basamaklı bir sayının basamaklarının küpleri toplamı sayının kendine eşit olup olmadığını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bu soruda ilk önce yapmamız gereken basamak değerlerini bulmaktır. Bunun için sayıyı devamlı 10’a bölmeliyiz her zaman kullandığımız gibi % işareti bölme işleminde kalanı veriyor / işareti ise bölümün tam kısmını veriyor. Üs alma işareti olarak da ^ işaretini kullanıyoruz. Soruda döngü işlemleri ve şart işlemleri ön plana çıkıyor.

**CEVAP-)**

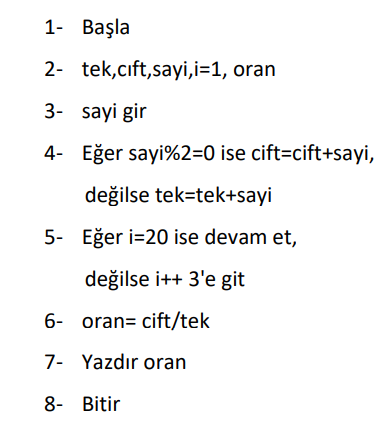
****

**SORU-)** Klavyeden girilen 20 adet sayıdan çift sayıların toplamının tek sayıların toplamına oranını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bu soruda döngü ve şart mekanizması yine önümüze çıkıyor. 20 defa sayı girmek yerine programda 20 defa çalışacak bir döngü kurmalıyız. Eğer mekanizması, (%) kalanın tek mi çift mi olduğunu saptar ve bunları boş değişkenlere atar. Kalan tek ise tek değişkenine, çift ise cift değişkenine toplanarak atılır (Bu değişkenlere akılda daha kolay tutulabilmesi için kese de diyebiliriz). Burada tek ve cift değişkenlerinin değerleri başta 0 olmak zorundadır. Buna dikkat etmeliyiz, yoksa cift=cift+sayi işlemini yapamayız. Bu sorunun algoritma testini 20 sayı için değil 4 sayı için yapıyoruz. Çünkü algoritmayı kurduğunuzda algoritma dinamik ise sayı 4 olsun 1004 olsun fark etmez. Bunu için statik çözümler yerine dinamik çözümler üretmeliyiz.

**CEVAP-)**

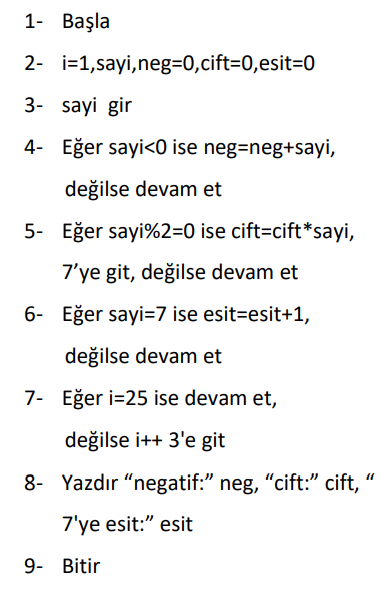
****

**SORU-)** Klavyeden girilen 25 adet sayı içerisinden negatif olanların toplamını, çift sayıların çarpımını, 7'ye eşit olanların adetini bulup ekrana yazdıran programın algoritma ve akış diyagramını çiziniz.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Bu soru basit olmakla beraber, programlama dilinde çok önemli ve çok kullandığımız if yapılarını anlatıyor. Tabiiki algoritmada if yerine eğer kullanıyoruz. Çünkü algoritmada program dili komutları kullanmıyoruz. Yine bir döngümüz var 25 tane sayı girdiriyor ve bu sayıları 3 şart ile kıyaslayıp hangisine uyarsa o değişkenin içine atıp toplattırıyoruz. Bu soru daha çok alıştırma sorusu niteliğindedir. Soru basittir ama bize eğer(karar) mekanizmasına iyice alışmamız gerektiğini anlatmaktadır. Buna örnek için içe for döngülerinde matris işlemlerini örnek verebiliriz. Bu testi yaparken programın dinamik olduğunu göz önüne alarak biz 5 sayı için yapmayı tercih ettik. İstenilirse 25 için de, 5 için de N kez için de olabilir.

**CEVAP-)**

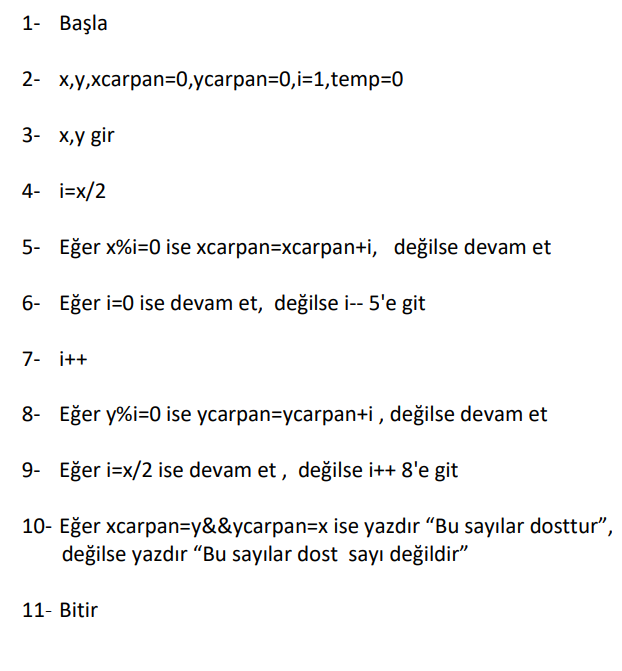
****

**SORU-)** .X, Y pozitif olmak üzere, eğer x sayısının çarpanları toplamı y sayısına ve aynı zamanda y sayısının çarpanları toplamı x sayısına eşit ise bu sayılar dost sayılardır. Buna göre girilen iki sayının dost olup olmadığını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz), Inheritance (Kalıtım), Döngü Koşul

**Açıklama:** Dost sayılar MÖ 500’lü yıllarda Pisagor okulunda sayı mistizmi üzerine araştırma yapan matematikçilerin yarattıkları bir sayı grubudur. Böyle soruları çözmek bizim algoritma konusunda gelişmemizi sağlar. Burada 2 adet sayaç kullanmamız gerekmektedir, sayaçlarımız ycarpan ve xcarpandır. Bu değişkenleri başta 0 olarak tanımlamalıyız. Buna dikkat edilmelidir. Örneğin eğer biz değişkenleri tanımlarken xcarpan=0 olarak tanımlamasaydık algoritmada 5. adımda bulunan xcarpan=xcarpan+i işlemi çalışmazdı i değeri ile hangi değeri toplayacağını derleyici bilemez ve hata döndürür. Burada 4. adımda X sayısını 2’ye bölmek de önemlidir. Çünkü bir sayıyı kendi hariç bölebilecek en büyük sayı yarısı kadardır. Örnek: 32 sayısını bölen, kendisinden hariç en büyük sayı 16’dır. Bunun için döngümüzü i=0 dan i=16 ya kadar götürmek hız açısından daha uygun olacaktır. Bu soru da öncelikle X sayısının tam bölenlerini bir değişkene atıyor, sonra Y değişkeninin bölenlerini bir değişkene atıyor. Sonra bu iki değişkeni ile karşılaştırıp eşitlerse ekrana “dost” ya da eşitsizliğe göre “dost değil” olarak yazdırıyor. Bu soru döngü ve karar mekanizmaları açısından bize idman yaptıran önemli bir sorudur.

**CEVAP-)**

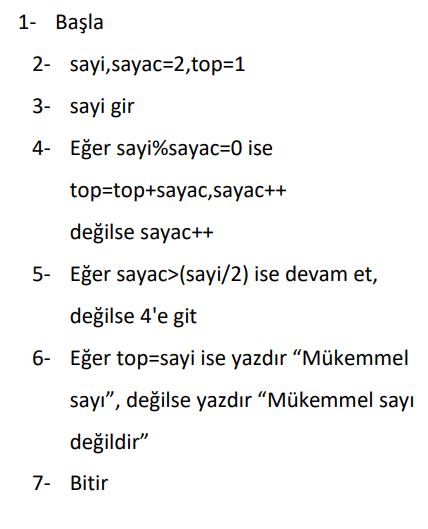
****

**SORU-)** Girilen sayının mükemmel sayı olup olmadığını bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz), Inheritance (Kalıtım), Döngü, Koşul

**Açıklama:** Mükemmel sayı sorusu bizim en sevdiğimiz soru çeşitlerindendir. Döngü kavramı karşımıza burada daha açık bir şekilde çıkmaktadır. Mükemmel sayı, kendisini tam bölen sayıların toplamı, kendine eşit olan sayılardır. Örnek 28’dir. 1+2+4+7+14 = 28

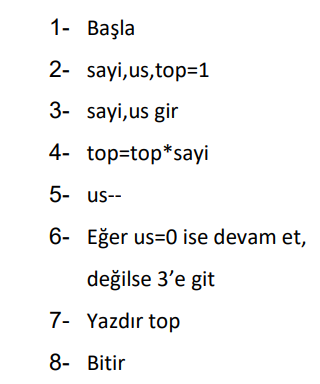
**CEVAP-)**

****

**SORU-)** Herhangi bir sayının herhangi bir dereceden kuvvetini bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Inheritance (Kalıtım), Koşul

**CEVAP-)**

****

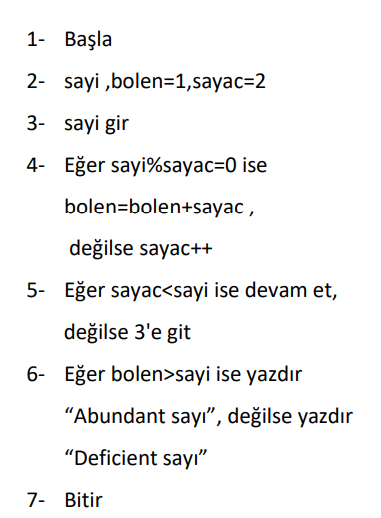
**Adınız Soyadınız:**

**SORU-)** Girilen sayının abundant (güçlü) sayı mı ya da Deficient (güçsüz) sayı mı olduğunu bulan programın algoritmasını yazınız.

**KULLANICAKLARINIZ-)** Interface (Ara Yüz), Inheritance (Kalıtım), Döngü, Koşul

**Açıklama:** Bu soru aslında mükemmel sayı mantığına yakın bir sorudur. Döngü ve karar mekanizmaları bu soruda da bulunmaktadır. Önemli olan programlamada if ve döngü yapılarıdır. Bunlar çok iyi kavranmalıdır. Bu soruda bir sayı giriyoruz. Bu sayıyı tam bölen sayıları bulmak için, sayı/2’ ye kadar sayımızı sayaca böldürüp kalan değer 0 ise bunu bölen adlı değişkenin içine toplayarak atıyoruz. Çıkan bölen değer ile sayıyı karşılaştırıp büyük ise “güçlü”, küçük ise “güçsüz” diye ekrana bastırıyoruz.

**CEVAP-)**

****