1. Bir problemi çözmek veya belirli bir hedefi gerçekleştirmek için gerekli olan mantıksal ve sıralı adımların tümüne algoritma denir. Programlamanın en önemli kısmı karşılaştığımız sorunların çözümü için algoritma geliştirebilmektir. Eğer algoritmayı düzgün bir şekilde hazırlarsak programı kodlamak çok daha basit hale gelecektir. Algoritma yapmak istediğimiz işi bütüncül bir şekilde değerlendirerek en basit şekilde tamamlanmasına yardımcı olur. Grup halinde çalışmayı mümkün kılar. Tüm aşamalar yazıldığı için sadece düşünen kişi değil algoritmaya sahip olan herkes proje üzerinde çalışabilir.

2. Akış diyagramı tamamlandıktan sonra üzerinde fazla değişiklik yapılmaması daha uygundur. Dolayısıyla önce algoritma daha sonra akış diyagramı hazırlanır.

3. Algoritmanın doğru bir şekilde geliştirilebilmesi için anlamamız ve uymamız gereken bazı kurallar vardır. Öncelikle bilgisayarların düşünemediğinin anlaşılması gerek. Dolayısıyla algoritmanın her adımının basit ve net bir şekilde ifade edilmesi gereklidir. Belirsizliğe ve yoruma açık olmamalıdır. Bir başlangıcı ve sonu olmalıdır. Sonsuz döngüye girmemelidir. Aynı işlemler yapıldığı takdirde her çalıştırma için aynı sonucu vermelidir. Giriş/Çıkış’a sahip olmalıdır. Giriş, programın üzerinde işlem yapacağı değerleri çıkış ise bu işlemlerin sonucunda üretilen sonuçları temsil eder.

4. Programlamada işlenen verileri bilgisayarın hafızasına tutmak için kullandığımız terimler değişken olarak bilinir. Değişkenler sayesinde aynı veriyi verdiğimiz isim ile çağırarak üzerinde defalarca işlem gerçekleştirebiliriz. Programın her çalıştırılmasında farklı değerler alabilir.

degisken\_ismi = Değer şeklinde tanımlanır

Değişken ismi kısmı değişkeni o satırdan sonra çağırabileceğimiz isimdir.

Değişkenin isminin içeriğini çağrıştırması programı daha anlaşılır hale getirecektir.

Değer kısmı matematiksel, mantıksal veya bir karakter ifade olabilir.

Değişkene yeni bir değer aktarırsak eski değer silinecektir.

5. Değişkenler kısmında değişkene yeni bir değer aktarıldığında eskisinin silindiğini söylemiştik. Bu prensibe göre bazı işlemlerin belirli sayıda yapılması ve sayılması için sayaçları kullanırız. Örneğin bir sayı grubunda kaç tane 5 olduğunu bulmak istersek tüm rakamları sırayla çağırıp 5 olup olmadığı sorgularız. Çağırdığımız rakam 5 ise bunu sayacak olan farklı bir değişkenin değeri her seferinde bir arttırılır.

6. 60

7. 70

8.

1. başla

2. İlk sayıyı (sayi1) girin

3. İkinci sayıyı (sayi2) girin

4. eğer sayi1=sayi2 ise git 3

5. Üçüncü sayıyı (sayi3) girin

6. eğer sayi3=sayi2 ya da sayi3=sayi1 ise git 5

7. eğer sayi2<sayi1 ve sayi3<sayi1 ise B=sayi1

8. eğer sayi1<sayi2 ve sayi3<sayi2 ise B=sayi2

9. eğer sayi1<sayi3 ve sayi2<sayi3 ise B=sayi3

10. Yaz B

11. Dur

9.

1. başla

2. İlk sayıyı (sayi1) girin

3. İkinci sayıyı (sayi2) girin

4. eğer sayi1=sayi2 ise git 3

5. Üçüncü sayıyı (sayi3) girin

6. eğer sayi3=sayi2 ya da sayi3=sayi1 ise git 5

7. eğer sayi1<sayi2 ve sayi1<sayi3 ise K=sayi1

8. eğer sayi2<sayi3 ve sayi2<sayi1 ise K=sayi2

9. eğer sayi3<sayi2 ve sayi3<sayi1 ise K=sayi3

10. eğer sayi1=K ve sayi3<sayi2 ise O=sayi3 B=sayi2

11. eğer sayi1=K ve sayi2<sayi3 ise O=sayi2 B=sayi3

11. eğer sayi2=K ve sayi1<sayi3 ise O=sayi1 B=sayi3

12. eğer sayi2=K ve sayi3<sayi1 ise O=sayi3 B=sayi1

13. eğer sayi3=K ve sayi1<sayi2 ise O=sayi1 B=sayi2

14. eğer sayi3=K ve sayi2<sayi1 ise O=sayi2 B=sayi1

15. Yaz K, O, B

16. Dur

10.1(Tek döngü, mod) 1 ve 99 dahil edilmemiştir.

1. Başla

2. tektop=0

3. cifttop=0

4. tekcarp=1

5. ciftcarp=1

6. sayac=2

7. eğer sayac>=99 ise git 13

8. kalan=sayac MOD 2

9. eğer kalan=0 ise cifttop=cifttop+sayac ve ciftcarp=ciftcarp\*sayac

10. değilse tektop=tektop+sayac ve tekcarp=tekcarp\*sayac

11. sayac=sayac+1

12. git 7

13. Yaz Tek Sayıların Toplamı (tektop), Tek Sayıların Çarpımı (tekcarp), Çift Sayıların Toplamı (cifttop), Çift Sayıların Çarpımı (ciftcarp)

14. Dur10.2(Ayrı Ayrı)

1. Başla

2. sabit\_tt=3

3. toplam\_tt=0

4. sabit\_ct=4

5. toplam\_ct=0

6. sabit\_tc=1

7. toplam\_tc=1

8. sabit\_cc=2

9. toplam\_cc=1

10. eğer sabit\_tt>=99 ise git 14

11. toplam\_tt=toplam\_tt+sabit\_tt

12. sabit\_tt=sabit\_tt+2

13. git 10

14. eğer sabit\_ct>=99 ise git 18

15. toplam\_ct=toplam\_ct+sabit\_ct

16. sabit\_ct=sabit\_ct+2

17. git 14

18. eğer sabit\_tc>=99 ise git 22

19. toplam\_tc=toplam\_tc\*sabit\_tc

20. sabit\_tc=sabit\_tc+2

21. git 18

22. eğer sabit\_cc>=99 ise git 26

23. toplam\_cc=toplam\_cc\*sabit\_cc

24. sabit\_cc=sabit\_cc+2

25. git 22

26. Yaz toplam\_tt, toplam\_tc, toplam\_tc, toplam\_cc

27. Dur