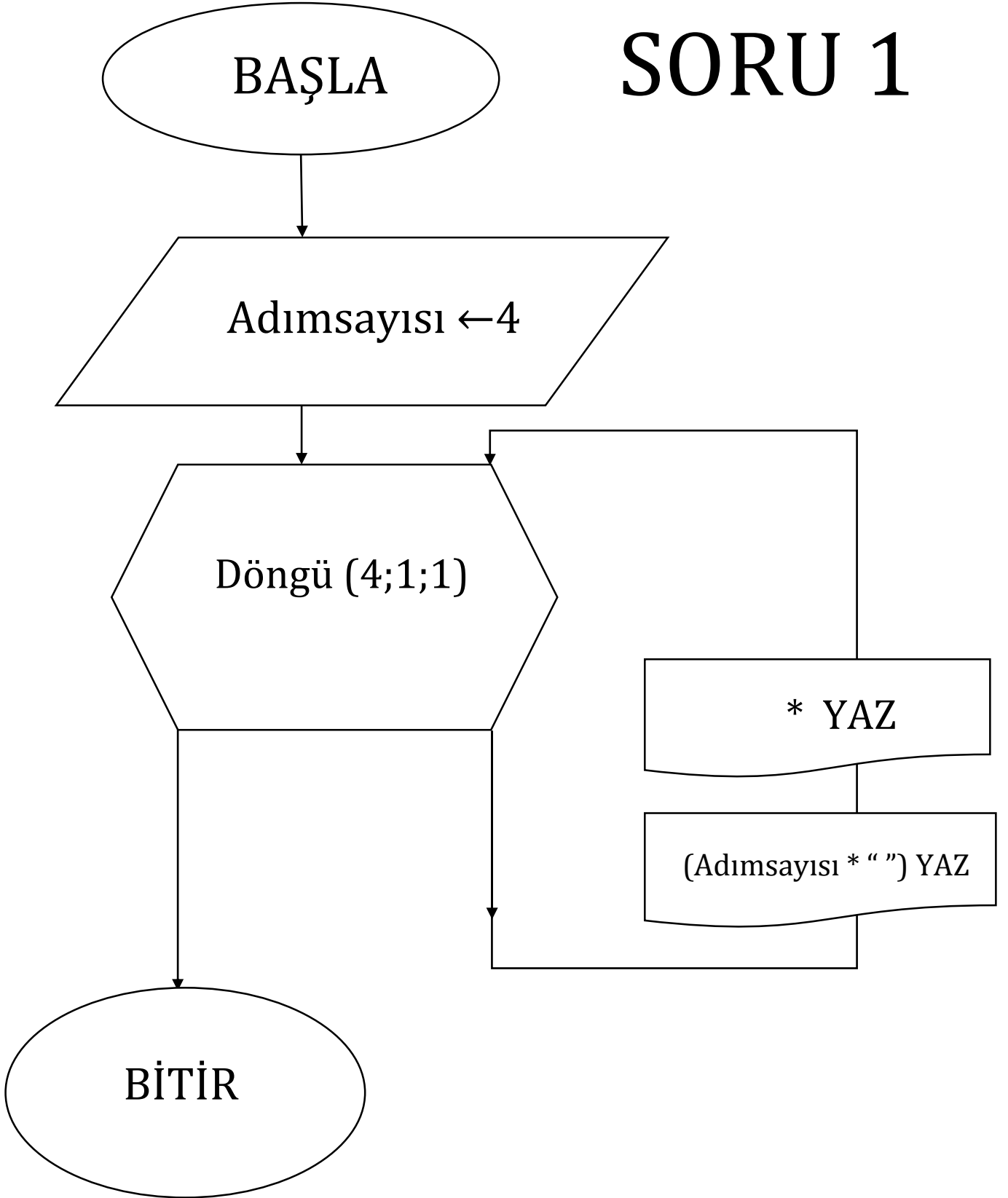


# SORU 1



## Soru 1'in Algoritması

1.BAŞLA

2.AdımSayısı=4 verisini oku

3.Döngü Başlangıcı= (AdımSayısı>=1) oldukça işlemleri tekrarla

4.AdımSayısı kadar (YAZ " ")

5. YAZ \*

6.AdımSayısı=AdımSayısı-1

7.Döngü Sonu

8.BİTİR

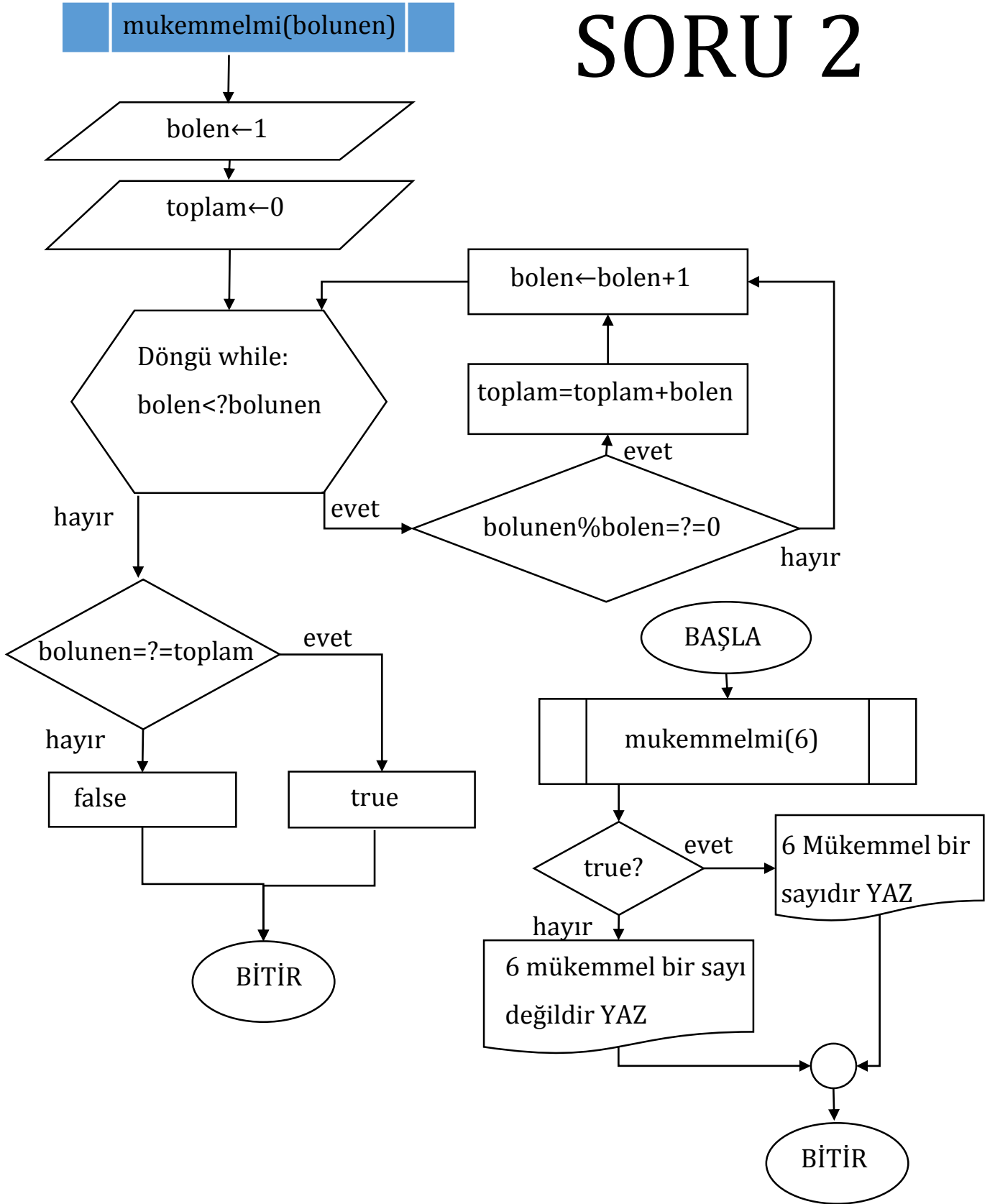
## Soru 1- Açıklama

Bu soruda önce "Adım sayısı" adında bir değişken tanımladık. Sonra "while" koşulu ile sıfıra eşit olana kadar ekrana değişkenimizin(Adım sayısı) sayısal değeri mukabilinde boşluk ve sonrada '\*' yazdırdık. Bu işlemi "Adım sayısı" 0'a eşit veya küçük olana kadar devam ettik. "Adım sayısı" değişkenimiz, 0 sayısal değerine eşit olduğu zamansa döngü sona erdi.

'Cygwin64 terminal' de soru1.rb'nin ekran görüntüsü:

```
User@15060033-dell /cygdrive/d/Taha Yasir/dersler/programlama dersi/ödevler/ödev 4/dosyalar
$ ruby soru1.rb
  *
 *
*
*
*
```

# SORU 2



1. BAŞLA
2. Mukemmelmi fonksiyonunu 6 parametresiyle başlat
3. Mukemmelmi fonksiyonu başlangıcı(bolunen parametresini OKU )
4. bolen=1
5. toplam=0
6. Döngü Başlangıcı : bolen<bolunen oldukça tekrarla
7. Eğer başlangıcı, eğer bolen, bolunenin tam katıysa  
toplam=toplam+bolen
8. Eğer Sonu.
9. bolen=bolen+1
10. Döngü sonu.
11. Eğer başlangıcı, eğer ( bolunen == toplam) ise RETURN(DÖN) true
12. (bolunen == toplam) değil ise RETURN(DÖN) false
13. Eğer Sonu.
14. Mukemmelmi fonksiyonu sonu.
15. Eğer Başlangıcı, (mukemmelmi(6) == true) ise “6 mükemmel bir sayıdır YAZ.
16. Eğer (mukemmelmi(6) == true) değil ise “6 mükemmel bir sayı değildir” YAZ
- 17.BİTİR

Kodun çalıştığına dair resim:

```
User@15060033-dell /cygdrive/d/Taha Yasir/dersler/programlama_dersi/ödevler/ödev
4/dosyalar
$ ruby soru2.rb
6 mükemmel bir sayıdır
```

	bölünen	bölen	toplam
Döngüye girmeden önce	6	1	0
While 1. Adımdan sonra	6	2	1
While 2. Adımdan sonra	6	3	3
While 3. Adımdan sonra	6	4	6
Döngüden çıktıktan sonra	6	4	6

Açıklama:

Öncelikle “mukemmelmi” adında bir fonksiyonu tanımlandı. Bu fonksiyonun içeriği şu şekilde:

Önce parametre halinde bölünen değeri istendi, sonra bölen ve toplamı tanımlandı. Sonra bölen, bölünenden küçük oldukça devam eden bir döngü tasarlandı. Bu döngünün içine bir “eğer” yapısıyla birlikte bölenin, bölünenin tam katı oldukça toplama katılmasını sağlayan bir yapı hazırlandı. Sonra döngü sona erdirildi. Fonksiyonun son işlevi olarakta eğer fonksiyonda bölünenler toplama eşitse ‘true’ cevabını verecek, yanlıssa ‘false’ değerini verecek bir “eğer” yapısı tasarlandı. Sonra döngü bitirildi. En son ise mukemmelmi fonksiyonu 6 parametresiyle çalıştırıldığı zaman doğruysa “6 mükemmel bir sayıdır”, yanlıssa “6 mükemmel bir sayı değildir” yazan bir “eğer” yapısı tanımlandı.

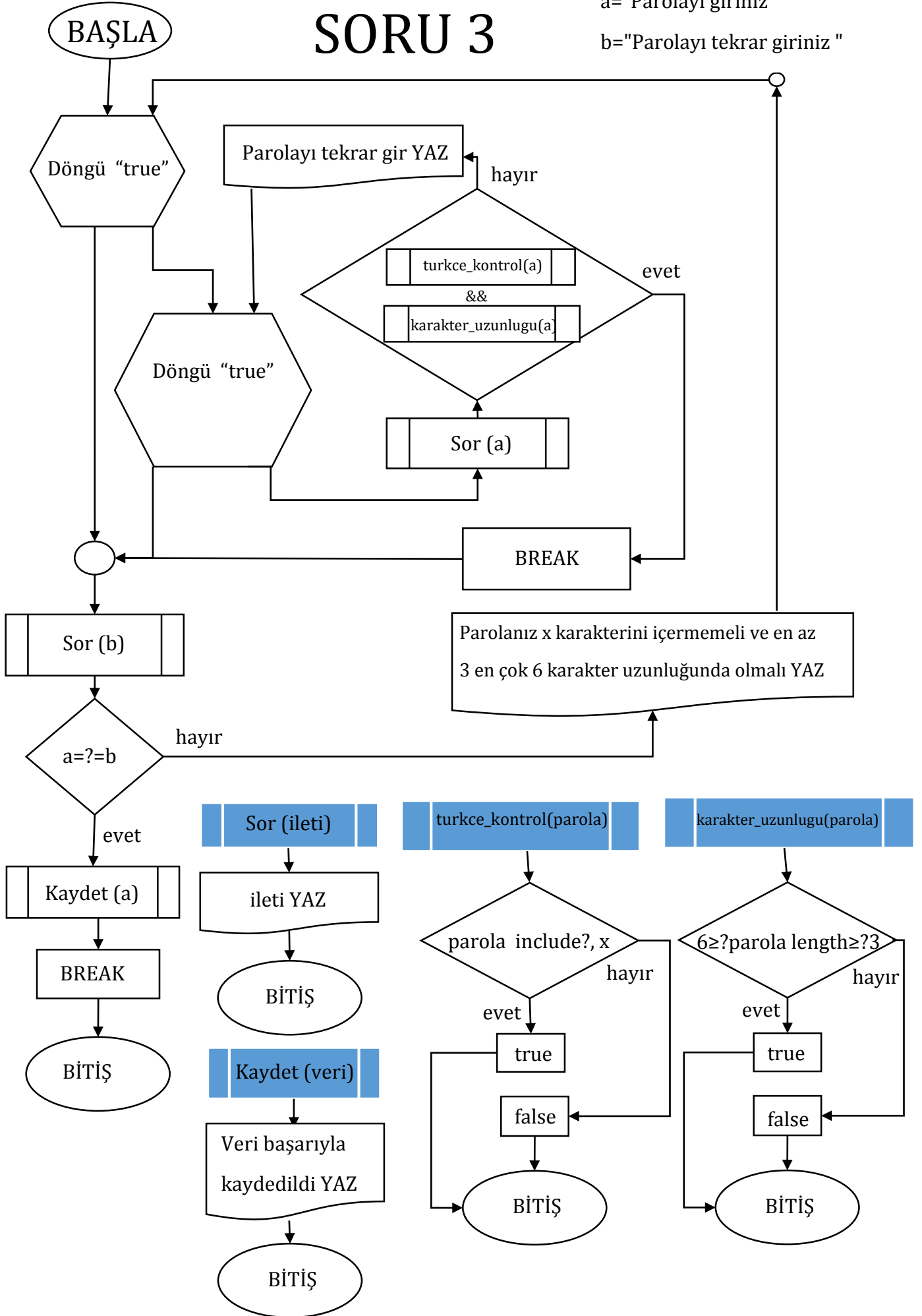
Normalde flowchartların bir bütün olması gerekir. Fakat ben hem mukemmel mı fonksiyonu bir fonksiyon olarak belirtmek hemde başlangıcın yalnızca fonksiyonun dışındaki kısım tarafından başlatılabileceğini anlatmak için ayırdım. Diğer bir deyişle bu iki yapıyı kodu iyi anlatacak şekilde birleştiremedim ve böyle bir yönteme başvurdum.

Bu kod yukarıdaki ekran görüntüsünde de görülebileceği gibi çalıştırıldığı zaman “6 mükemmel bir sayıdır” cevabını döner.

# SORU 3

a="Parolayı giriniz "

b="Parolayı tekrar giriniz "



## Algoritma


1. Başla
2. Döngü1'in başlangıcı ; Sonsuz Döngü
3. Döngü2'nin başlangıcı ; Sonsuz Döngü
4. Sor fonksiyonunu girilen parametreye göre çalıştır ve cevabı a'ya ata
5. Eğer "turkce\_kontrol" ve "karakter\_uzunlugu" fonksiyonlarını adım 5 de alınan a parametrik değerine göre çalıştıgında her ikisinde "true" değerini veriyorlarsa, Döngü2'yi kır (BREAK [ 10.adıma git] )
6. Değilse ekrana "parolanız 'x' karakterini içermemeli ve en az 3 en çok 6 karakter uzunluğunda olmalıdır" YAZ
7. 3.Adıma git
8. Eğer sonu
9. Döngü2 sonu
10. Sor fonksiyonunu girilen parametreye göre çalıştır ve cevabı b'ye ata
11. Eğer (a==b) ise kaydet fonksiyonunu a parametresine göre çalıştır
12. Döngü1'i kır (BREAK [17. adıma git])
13. Değilse "Yanlış parola girdiniz. Lütfen işlemi tekrar ediniz" YAZ
14. Adım 2 'ye git
15. Eğer Sonu
16. 1.Döngü Sonu
17. BİTİR

Kodun ekran görüntüsü:

```
Parolayı giriniz : xavier
Parolanız 'x' karakterini içermemeli ve en az 3 en çok 6 karakter uzunluğunda ol
malı.
Parolayı giriniz : y
Parolanız 'x' karakterini içermemeli ve en az 3 en çok 6 karakter uzunluğunda ol
malı.
Parolayı giriniz : tahayas
Parolanız 'x' karakterini içermemeli ve en az 3 en çok 6 karakter uzunluğunda ol
malı.
Parolayı giriniz : yasir
Parolayı tekrar giriniz : yasir
Seçmiş olduğunuz 'yasir' parolanız başarılı bir şekilde kaydedildi.
=> nil
```

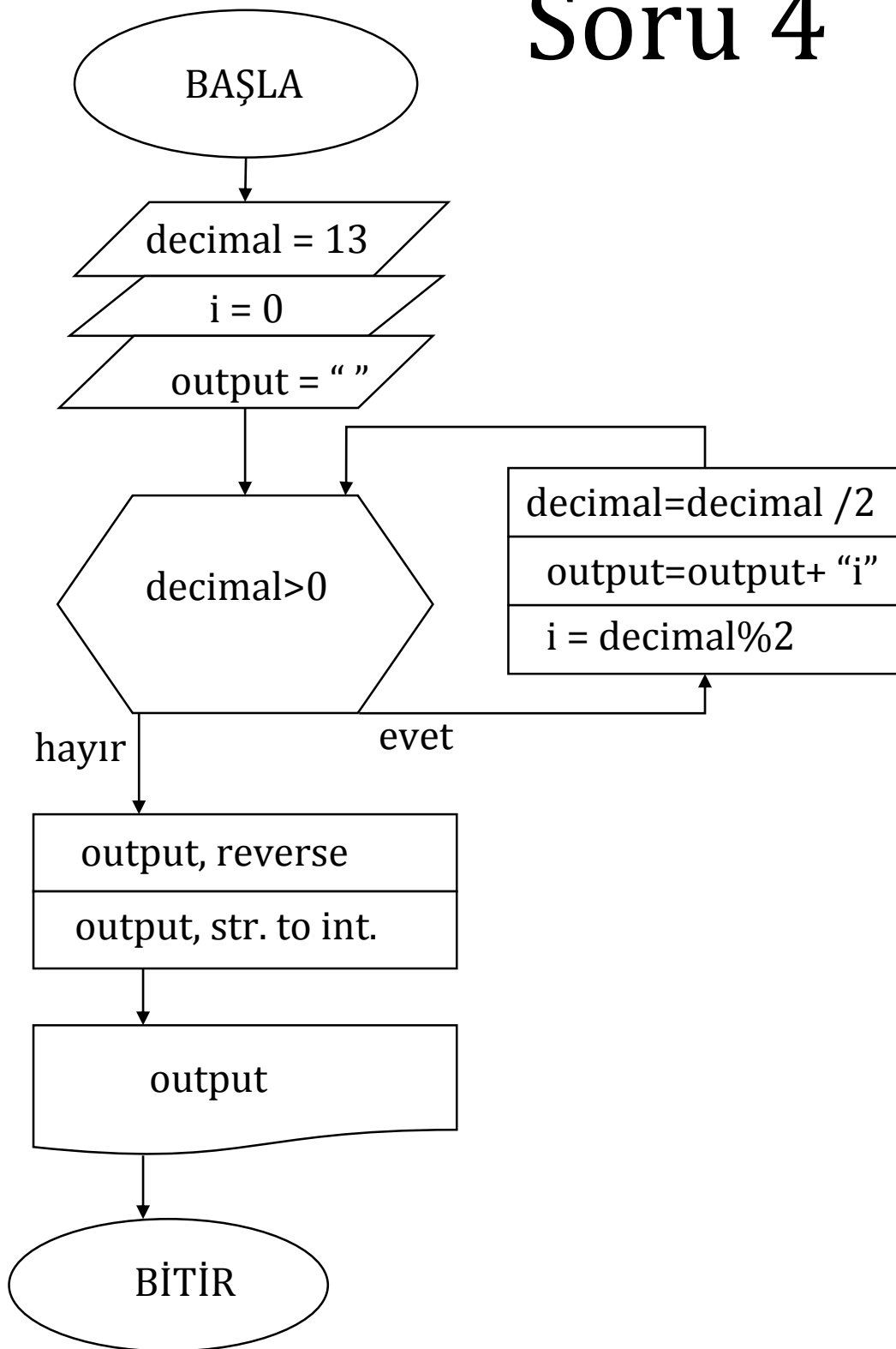
## Açıklama:

Bu programda ilk önce “kaydet”, “sor”, “turkce\_kontrol” , “karakter\_uzunlugu” adlarında 4 farklı fonksiyon oluşturduk. Sonra girilen parola’nın yanlış olma ihtimaline karşı iki döngü oluşturduk. İç döngü, karkterler arasında x harfinin olup olmadığı ve parola’nın 6 ve 3 kapalı aralığında olup olmadığına bakıyor. Dış döngü ise ilk girilen ve ikinci girilen parolaları kıyaslayarak kaydedilecek verinin hata payını azaltıyor. En son ise kaydet fonksiyonu çalışarak parolanın kaydedildiği hakkında ekrana bilgi çıktısı veriyor.

Flow chartlarda mavi ile başlayan “” şeklinde olan şekiller fonksiyonları tanımlamak için kullanılmıştır. Bunların başına kasıtlı olarak “başla” komutu konulmamıştır. Sadece çağırıldıkları zaman çalışırlar. Bunu bu şekilde yapmamın sebebi fonksiyonları yerine yerleştirmem halinde şuanda olduğundan çok daha karışık bir hal alıyor olmasıdır.



# Soru 4



#### Soru 4'in Algoritması

1.BAŞLA

2.decimal=13

3.i=0

4.output= " "

5.Döngü başlangıcı= (decimal>0) oldukça işlemleri TEKRARLA

6.i=decimal%2(decimal mod2 'ye göre yazıldığı zaman kalan)

7.output=output+"i"

8.decimal=decimal/2

9.Döngü sonu

10.output'un elemanlarını TERSİNE ÇEVİR (REVERSE)

11.output'un türünü sayıya DÖNÜŞTÜR

12.YAZ output

13.BİTİR

#### SORU-4 için Açıklama

Bu soruda amaç 10 tabanındaki (decimal) bir sayıyı, 2 tabanındaki(binary) bir sayıya çevirmek. 10 tabanını 2 tabanına çevirmek için önce "decimal" adında bir değişken atadım. Bu değişkene bize hesaplatılmak istenen '13' integer verdim. Sonra "i" adında başka bir değişken atadım. Bu değişkenin amacı decimal sayının 2 tabanında yazıldığında, birler basamağında kalan sayıları tutmak ve daha sonra bunu işleyecek olan "output"a yazmak. En son olarak "output" değişkenini atadım. "output"u atamamdaki amacım decimal'ı binary tabanına çevirirken birler basamağında kalan sayıları yanyana dizmek. Topladığımız bu elemanlar binary tabanının tersten yazılmışa eşdeğer olacağı için 10.adımda elemanlarını ters çevirdik ve binary hâlini elde etmiş olduk.

Döngü içinde yaptığımız işlem, decimal sayıyı 0'a eşit olana kadar mod 2'ye göre yazmak ve kalanları toplayarak binary sayımızı elde etmek.(döngüde decimal'ın 0'a eşit olmamasının sebebi kod'u sonsuz döngüye sokacak olması ve binary'nin değerini sonsuz olarak arttıracak olmasıdır.)

	decimal	i	output
Döngüye girerken	13	0	-
1.while'dan sonra	13	1	1
2. while'dan sonra	6	0	10
3.while'dan sonra	1	1	101
4.while'dan sonra	0	1	1011
Döngüden çıktıktan sonra	0	1	1101

“output” döngüye girerken “” string değerinde olduğu için hiçbir değeri yoktur ve ‘output.empty?’ komutuna karşı ‘true’ değerini verir.

Cygwin64Terminal’de kodun ekran görüntüsü:

```
User@15060033-dell /cygdrive/d/Taha Yasir/dersler/programlama_dersi/ödevler/ödev
4/dosyalar
$ ruby soru4.rb
1101
```