

OLC731
R YAZILIMI ILE VERİ ANALİZİ

Kubra Atalay Kabasakal

2023-10-23

Contents

Ders Hakkında

Bu dersin amacı, R yazılımı kullanarak veri üzerinde istenilen çok değişkenli istatistiksel ve psikometrik işlemlerin yapılabilmesini sağlamaktır.

Tez ve makale çalışmalarında öğrencilerimizin analizlerini R yazılımı ile hiçbir paket programa ihtiyaç duymadan kendi başlarına yapmalarını sağlamaktır.

(zence) **zence**.

Eğitmen

👤 Dr. Kübra Atalay Kabasakal

✉ katalay@gmail.com

✉ katalay@hacettepe.edu.tr

Kitaplar

- Atar, B., Atalay Kabasakal, K., Unsal Ozberk, E. B., Ozberk, E. H. & Kibrislioglu Uysal, N. (2020). R ile Veri Analizi ve Psikometri Uygulamalar, Pegem Akademi, Ankara.
- Desjardins, C. D., & Bulut, O. (2018). Handbook of educational measurement and psychometrics using R. Boca Raton, FL: CRC Press.
- Demir, E. R Diliyle istatistik Uygulamalar. Pegem Akademi, Ankara.(2021).

Kaynaklar

- Bu ders materyallerine ek olarak, R öğrenmek için bir dizi mükemmel kaynak vardır:
- R kurulumu ile ilgili bilgiler içerir.
- R studio ve güncellemeler ile ilgili bilgiler içerir.
- – Alana ozgu paketlerini inceleyebilirsiniz
- R Cookbook
- Veri Bilimi için R
- Hadley Wickham
- StackOverflow soru-cevap platformu
- Twitter’da #rstats hashtag’ini arayın veya kullanın.
- e-posta gruplarındaki e-postalara `RSiteSearch ("sample.int")` "

- R ile ilgili farklı ücretsiz kaynaklar bulabilirsiniz (kitaplar, videolar, websiteleri). Bu kaynakların bazıları başlangıç seviyesinde.
- Referans kartları
- Cheat Sheets

Chapter 1

R ve Rstudio Temeller

Bu bölümde, **R** ve **RStudio**'nın nasıl kullanılacağını yanı sıra bazı temel programlama kavramları ve terminolojisi, yaygın tuzaklar, faydalı ipuçları ve nereden yardım alınabileceği konularını ele alacağız. Programlama deneyimi olmayanlar bu bölümü özellikle yararlı bulacaklar, ancak daha önce R kullanmış olsanız bile bazı yararlı ipuçları ve püf noktalar bulabilirsiniz.

Bu bölüm kendi kodunuzu yazmaya başlayana kadar bir anlam ifade etmeyebilir :)
Biraz sabretmenizi bekliyorum !

1.1 R Nedir?

- R istatistiksel hesaplamalar yapabilen bir programlama dilidir.
- 1996 yılında Auckland Üniversitesi'nde **Ross Ihaka** ve **Robert Gentleman** tarafından geliştirilmiştir.
- 1960 yılında Bell Laboratories'de John Chambers ve arkadaşları tarafından geliştirilen **S dilinin** açık kaynak kodlu halidir.
- R yazılım Genel Kamu Lisansı (GNU* General Public Licence) koşulları altında ücretsiz dağıtılmaktadır.
- R ve Temel Geliştirme Takımı (R core team) ile ilgili bilgilere R'in internet sitesinden (<https://www.r-project.org/>) ulaşılabilir.
- R dilinin ilk sürümü 29 Şubat 2000 tarihinde yayınlanmıştır. Her iki-üç ayda bir sürümler güncellenmektedir.
 - **R version 4.3.1 (Beagle Scouts)** has been released on 2023-06-16.
- RStudio, R ile çalışmayı kolaylaştıran bir Entegre Geliştirme Ortamıdır (Integrated Development Environment [IDE]).
- Bunu en azından bilmek ve kitap yazmak için Notepad gibi düz bir metin editörü kullanmak ile Microsoft Word gibi bir kelime işlemci kullanmak gibi düşünün. Bunu yapabilirsiniz, ancak bu kadar iyi görünmez ve yazım denetimi ve biçimlendirme gibi şeyler olmadan çok daha zor olur. Benzer bir şekilde, **R Studio olmadan da R kullanabilirsiniz ancak bunu tavsiye etmiyorum.**
- Unutmamanız gereken en önemli şey, bu ders için tüm çalışmalarınızı RStudio kullanarak yapacak olsanız da, aslında iki yazılım parçası kullanıyorsunuz, bu da zaman zaman her ikisinin de **ayrı güncellemeleri** olabileceği anlamına geliyor.

- R’yi ölçme için kullanmanın iki yolu vardır. İlk olarak, web tarayıcınız aracılığıyla R ve R’nin çevrimiçi bir sürümünü kullanabilirsiniz (**R server/sunucusu**). İkincisi, R ve RStudio’yu dizüstü veya masaüstü bilgisayarınıza ücretsiz olarak indirip kurabilirsiniz.

1.2 Avantajlar

- R özgür istatistiksel bir programlama dilidir.
- R aynı zamanda bir yorumlayıcıdır (interpreter).
- R, bir veri tabanı **deildir** ama veri tabanlarına bağlanabilir.
- Önceki sürümleri kullanıcı dostu olmasa da, son zamanlarda kod editörlerine çok sayıda ilave eklenmiştir.
- Ayrıca Java gibi dillerin araçları ile arayüz desteğine sahip bir yazılım geliştirme ortamıdır.
- Tablolardan oluşan yazılım paketlerine (Excel, Minitab gibi) benzemekle birlikte, yeni geliştirilen bazı paketler farklı arayüzler sağlamaktadır.
- Ücretsiz olması nedeniyle, ticari destek tabii bir yazılım değildir. Ancak destek alınabilecek çok sayıda kaynağa erişilebilir. (stackoverflow, mail listeleri)

1.3 Neden R?

- R istatistiksel programlama, veri analizi ve grafiksel gösterim için kullanılan ve ticari bir amaç gütmeyen ücretsiz bir yazımdır.
- R, UNIX, Windows ve MacOS gibi çeşitli platformlarda kodlar derlemekte ve çalışmaktadır.
- SPSS, SAS gibi veri analizi programları ücretlidir, ayrıca bazı özel psikometri analizleri için yeterli değildir.
- R, açık kaynak kodlu olduğu için program kodlarına istenildiği zaman erişilebilir.
- Diğer istatistiksel yazımlar ile karşılaştırıldığında R **komut satırı** arayüzünü kullanmaktadır.
- **Basit kodlar, döngüler ve kısa tanımlı fonksiyonlar** yazmaya uyumlu basit ve etkili bir yazım diline sahiptir.
- R’in ayrıca **grafiksel imkânlar** oldukça fazladır; bu nedenle yayınlanabilir/basma uygun grafikler oluşturmak kolaydır.
- R ekibi birçok alanda ayrıntılı dokümantasyonu olan R paketleri geliştirmektedir.
- Klasik istatistik yazımlarında analiz sonuçları bir kez elde edilir. R yazılımında ise sonuçlar çalışma alanına kaydedilerek, ileriki analiz aşamalarında tekrar kullanılabilir.
- R, psikometri alanında sıklıkla kullanılan simülasyon çalışmaları için (tekrarlı işlemler için) de avantaj sağlamaktadır.
- R, diğer programlama dilleri ve istatistik paket programları ile uyumludur.

1.4 Dezavantajlar

- Basta öğrenilmesi kolay görünse de, R’da uzmanlaşmak oldukça zordur.
- Menü ile kullanılan programlara alışkın olan kişiler için başlangıçta korkutucu olabilir.
- R ile bir analizi yapabilmek için planlama yapılması gerekmektedir.
- R kullanıcıları çoğunlukla programlamacı **deildir**. Programlamaya hâkim olmayan kişiler tarafından hazırlanan, okunması ve sürdürülebilirliği zor kodlar oluşturulabilir.

- Balançta kodlar yazmak yldrc olabilir; ancak çalmaların tekrarlanabilirlii açsndan avantaj salamaktadır.
- Bu duruma bir örnek vermek gerekirse, 20 adet regresyon denklemi kurulup regresyon katsayılar karlatılmak istenirse, R yazlm sadece regresyon kat sayıların gösterebilir ve tek bir satrda tüm regresyon sonuçların karlatırmaya olanak verir. Aynı işlem için diğer istatistiksel yazımlarda 20 ayrı pencereden elde edilen sonuçların elle yazılarak karlatılması gerekecektir.
- R’da hata yapma olasılığı diğer programlara göre daha fazladır. Hata kaynağı için varsayımların iyi bilinmesi gerekmektedir.
- Hz konusunda SPSS ve SAS’a göre avantajlı olsa da diğer dillere göre (Python, Matlab gibi) daha yavaştır.
- Gelitirilen çok fazla paket olduğu için, ihtiyaca uygun en iyi paketin seçimi zor olabilmektedir.
- Bu bir dezavantaj gibi görünse de istatistiksel işlemlerin arka planı anlamaya yardımcı olur.
- Bu tarz zayıf hazırlanmış kodlar farklı koşullarda yavaş çalışabilmektedir.
- Çocu kullanıc bu eksiklikleri gidermek için kodları deitiremez. Özellikle çok iyi yapılandırılmamış olan kodlar R’da yavaş çalışabilmektedir.

1.5 R ve Rstudio Yüklenmesi

- İnternet tarayıcısına R yazılımının internet sitesinin ana sayfasının adresi yazılır. <https://www.r-project.org/>
- Sol menüde yer alan “download CRAN” bölümüne tıkladıktan sonra ülke seçilir. Seçilen ülkenin yakını sadece yükleme hızını deitirecektir.
- Çıkan sayfada “Download and Install R” başlığı altında iletim sistemine uygun olan bağlantı seçilir.
- R konsolda çalışmaya doğrudan bağlanabilir; ancak konsol kullanımı bir kod editörü olmadık için çok kullanışlı değildir.
- Rstudio hata ayıklama, görselleştirme araçları ile birlikte yüklenen modern bir kod editörüdür.
- <https://www.rstudio.com/> internet sitesinden kullanılan bilgisayar ve iletim sistemine uygun olarak seçilip indirilebilmektedir.
- Rstudio R ile daha üretken olmanıza yardımcı olacak bir dizi araç içerir, örneğin:
 - R kodlarının vurgulamak için bir sözdizimi vurgulama düzenleyicisi
 - R kodların yazılmanıza yardımcı olacak işlevler (otomatik tamamlama)
 - Çeşitli grafikler oluşturmak ve kaydetmek için çeşitli araçlar (ör. histogramlar, dalm grafiği)
 - Verileri içe veya dışarı aktarmak için bir çalışma alanı yönetim aracı

1.6 Diğer Gerekli Yüklemeler

- Benim açıklamaların yetmediyse R’yi bilgisayarınızda kullanmak için, lütfen daha ayrıntılı talimatlar ve indirmeniz gereken dosyaların bağlantılarını yanısıra R’yi farklı bilgisayar türlerine yüklemek için bir dizi klavye bağlantıları içeren Installing R adresine bakın!!
- Yüklemeler konusunda daha fazla ihtiyacınız var ise R studio R !
- Verilen linkte yer alsa da ayrıca eklemeye ihtiyaç duyduğunuz bağlantılar:
- Java
- Rtools Rtools, kaynak koddan derleme yapmaya yarayan araçlar içeren bir R yardımcıdır. **Önemli:** Eğer Windows kullanıyorsanız, ayrıca Rtools yüklemeniz gerekir.
- devtools

```
install.packages("devtools")
```

1.7 R STUDIO

- Rstudio'da panellerin yerlerini deitirebiliriz.
- Bunun yan sra yaz tipi, büyüklüü gibi özellikleri de deitirebiliriz.
- Varsaylan olarak, R Studio'yu açtınızda, kodunuz ve oluturduunuz tüm nesneler dahil olmak üzere en son ne üzerinde çalıştığınız gösterir. Bu yararlı gibi görünebilir, ancak aslında deerinden daha fazla soruna neden olma eilimindedir, çünkü yanlışlıkla bir nesnenin eski bir sürümünü kullanma riskiniz oldu anlamna gelir. R Studio'yu her balattınızda yeni bir kopya açacak ekilde ayarlar deitirmenizi öneririz. Bunu **Araçlar - Global Seçenekler** seçeneine tıklayarak ve ardından aadaki gibi görünmesi için kutuların seçimini kaldırarak yapabilirsiniz.

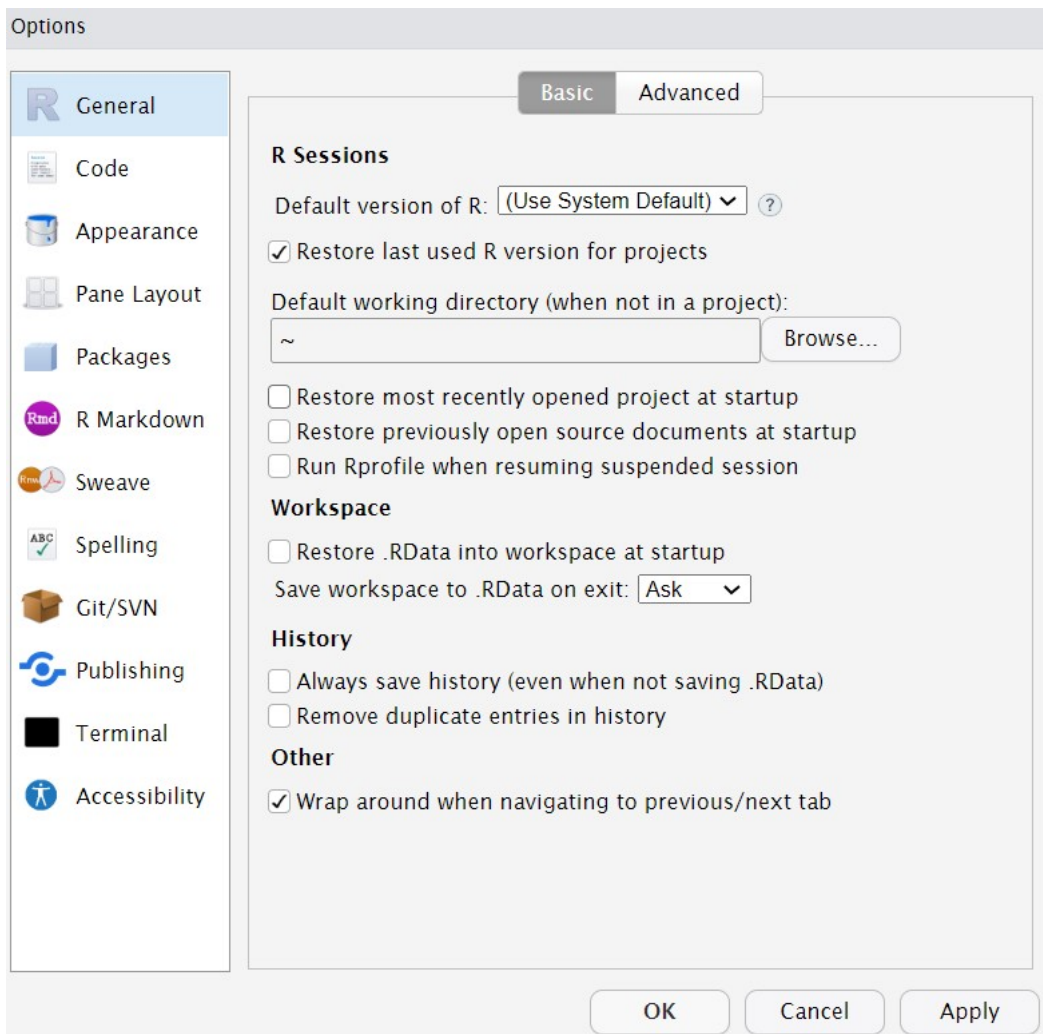


Figure 1.1: Global options

- Dönem boyunca Rstudio kullanıma aına olacaksınız. Bu süreci kolaylatırmak için balantlar verilen dökümanlara göz atabilirsiniz.
- Rstudio cheatsheet

- Oscar Torres* Reyna tutorial

1.8 Hangi R sürümünü kullanmalısınız?

- R'yi bilgisayarınıza kurmanın avantajı, kullanmak için internete bağımlı olmanız gerekmemesi, dosyalarınız kaydetmenin ve yönetmenin daha kolay olması ve sunucunun çökmesi durumunda sorun yaşanmamasıdır (bu nadirdir, ancak olmutur).
- R sunucusunu kullanmanın avantajı, bilgisayarınıza herhangi bir şey yüklemenize gerek olmaması, sadece web tarayıcınız üzerinden erişilebilirliğidir.
- R'yi yükleyemeyeceğiniz bir bilgisayarınız varsa (örneğin Chromebook) veya R'yi bilgisayarınıza yüklemeye ilgili ciddi sorunlarınız varsa sunucuyu kullanmanız önerilir.

1.9 R Studio Hakkında

- R Studio, kodu deneyebileceğiniz bir konsola sahiptir (ekil'de sol alt pencerede yer alır??).
- Ayrıca kod editörü (sol üst), "Ortam" sekmesinde oluşturduğunuz fonksiyonlar ve nesneleri gösteren bir pencere (sağ üst pencere) ve grafikleri, dosya paketlerini ve yardım belgelerini gösteren bir pencere (sağ alt) bulunur.

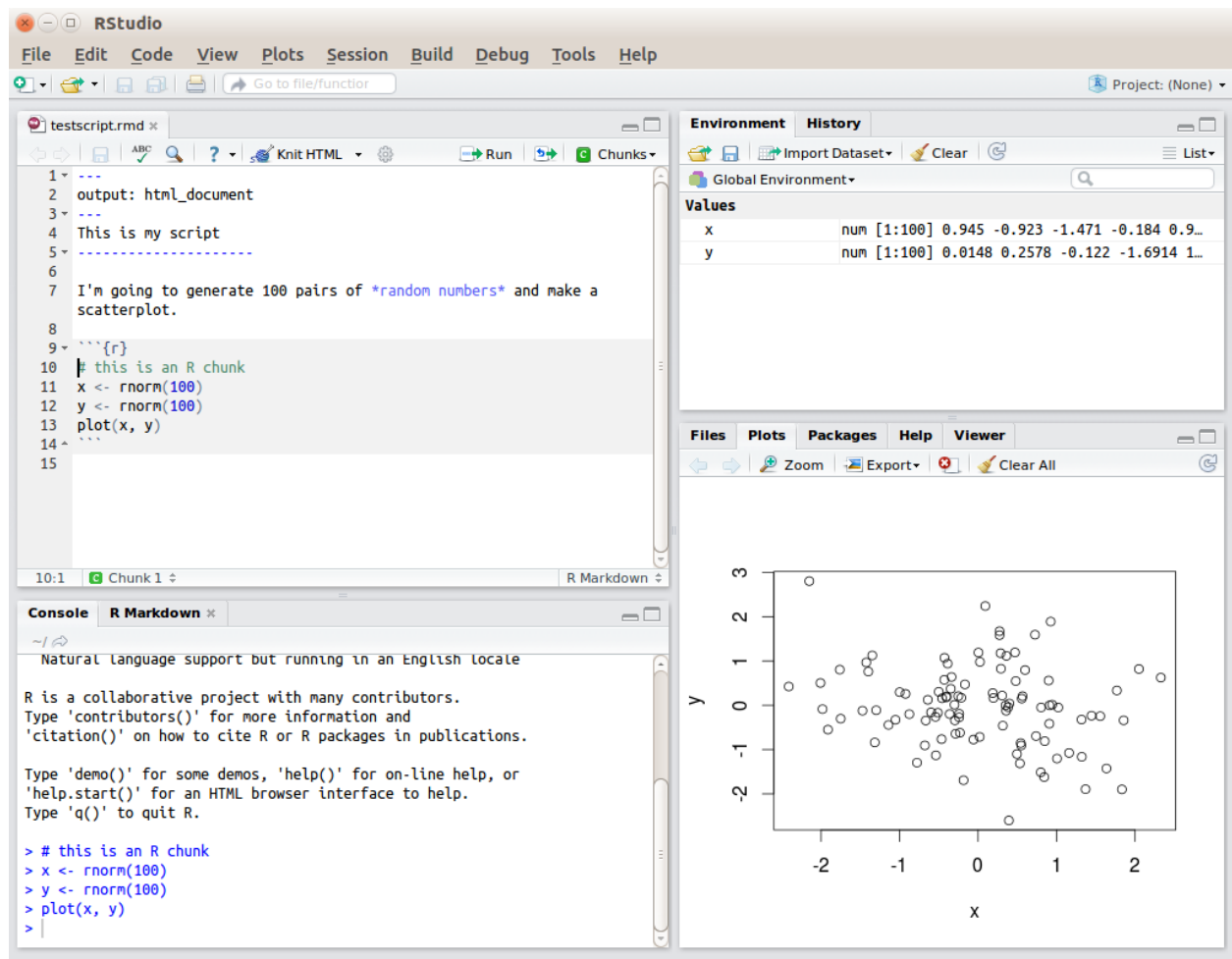


Figure 1.2: RStudio arayüzü

- Bu ders boyunca R Studio’da bulunan özelliklerin nasıl kullanılacağı hakkında daha fazla bilgi edineceksiniz, ancak R Studio ekibinden RStudio Essentials 1 izlemenizi ıddetle tavsiye ederim. Video yaklaşık 30 dakika sürmekte ve R Studio’nun ana bölümlerini tanıtmaktadır.

1.10 R Temel Özellikler

- R konsolda görünen `>` iareti, R yazılımınızdan komut beklediğini belirtir. R’in hesap makinesi olarak kullanımı örnekleri sunulmuştur.

```
2+2
```

```
[1] 4
```

- R bölümlere duyarlı değildir.

```
2 +      2
```

```
[1] 4
```

```
2+
2
```

```
[1] 4
```

1.11 Atama Operatörü

- Atama operatörü olarak “küçüktür” simgesi ile “kasa çizgi” simgesi `<-` simgeleri kullanılabilir.
- `<-` yerine “eittir” = simgesi de atama operatörü olarak kullanılabilir.
- Ancak `=` operatörü programlama yaparken matematiksel eşitlikle karabilmektedir.
- Atama yapılacak nesne isimlendirilirken harflerle (`A* Z` veya `a* z`) balamalıdır.
- simlendirmeye harfle balandıktan sonra rakamlar (`0* 9`), nokta (`.`) ve alt çizgi (`_`) ile devam edilebilmektedir.
- R harflerin küçük ve ya büyük olmasına kar duyarlıdır.
- R fonksiyonlarına benzer isimlerde nesne ismi kullanmamaya **dikkat edilmelidir**.
- Ayrıca `c,C,D,F,I,q,t,T` gibi tek harfli nesne ismi kullanmaktan kaçınılmalıdır; bunların R’da özel anlamlar bulunmaktadır.
- R yazılımında `#` iareti ile balayan satır, yorum satırdır.
- Genellikle komutların anlamını açıklamak için kullanılmaktadır.
- R, bu satırları dikkate almaz, bunları sadece kullanıcılar için bilgi ve hatırlatıcı açıklamalar içermektedir.

```
# Yorum satırları kodlarınızı anlamlı hale getirir.
```

```
a <- 2
```

```
y <- a * a
```

```
y
```

```
[1] 4
```

1.12 Basit İşlemler

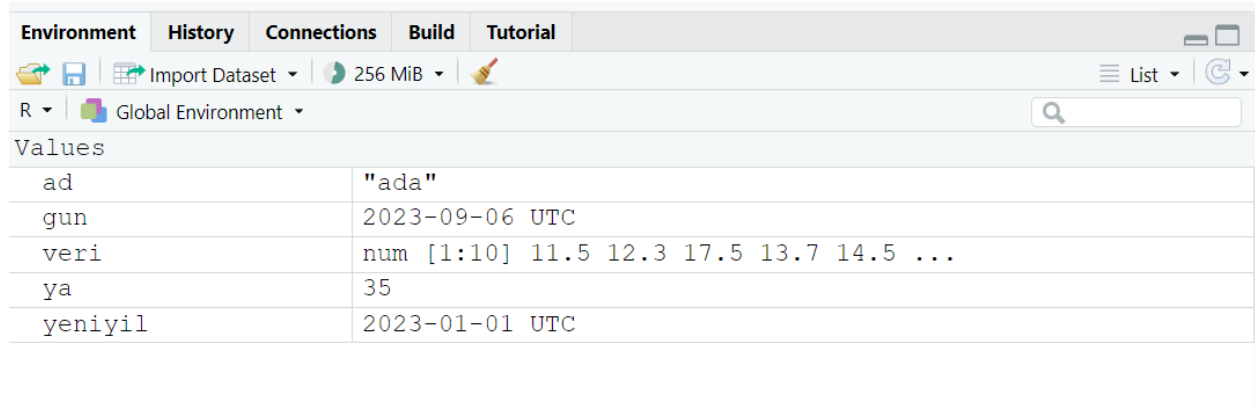
- toplama işlemi için `+`,
- çıkarma işlemi için `-`,

- çarpma ilemi için `*`,
- bölme ilemi için `/`,
- üs alma ilemi için `^` veya `*`
- mod alma için ise `%%` operatorleri kullanılmaktadır.
- Kodlamanın büyük bir kısmı nesne oluşturmayı ve nesneleri manipüle etmeyi içerecektir. Nesneler bir şeyler içerir. Bu şeyler sayılar, kelimeler veya işlemlerin ve analizlerin sonucu olabilir

Altmı Nesneler oluşturma

- Aşağıdaki kodu kopyalayıp konsola yapıştırın, kodu kendi adınız ve yanz kullanacak şekilde değiştirin ve çalıştırın. Environment bölümünde `ad`, `yas`, `gun`, `yeniyil` ve `veri` nesnelerinin görüldüğünü göreceksiniz.

```
ad <- "ada"
yas <- 16 + 20
gun <- Sys.Date()
yeniyil <- as.Date("2024-01-01")
veri <- rnorm(n = 10, mean = 15, sd = 3)
```



The screenshot shows the RStudio Environment pane with the following objects:

Object	Value
ad	"ada"
gun	2023-09-06 UTC
veri	num [1:10] 11.5 12.3 17.5 13.7 14.5 ...
ya	35
yeniyil	2023-01-01 UTC

Figure 1.3: Çalışma alanındaki nesneler

- Bu örneklerde, `ad`, `yas` ve `yeniyil` her zaman `ada`, `36` deerlerini ve `2024 Yeni Yıl Günü` tarihini içerecektir, ancak `gun` tarihi iletim sisteminden alacaktır ve `veri` rastgele oluşturulmuş bir veri kümesi olacaktır, bu nedenle bu nesnelerin deerleri statik olmayacaktır.
- Daha da önemlisi, nesneler hesaplamalara dahil olabilir ve birbirleriyle etkileime girebilir. Örnek:

```
yas + 10
yeniyil - gun
mean(veri)
```

```
[1] 46
Time difference of 70 days
[1] 14.34768
```

- Son olarak, bu işlemlerin sonucunu yeni bir nesnede saklayabilirsiniz:

```
n1 <- yas + 10
```

`<-` ifadesini ikiye bölerek okumak faydalı olabilir, örneğin `ad <- "ada"` ifadesi `ad` metnini içerir.

- Bu ders boyunca sürekli olarak nesneler yaratacaksınız ve ilerledikçe onlar ve nasıl davrandıkları hakkında daha fazla bilgi edineceksiniz, ancak imdilik bunların değerleri kaydetmenin bir yolu olduğunu, bu değerlerin sayı, metin veya işlemlerin sonucu olabileceğini ve yeni değişkenler oluşturmak için başka işlemlerde kullanılabileceğini anlamak yeterlidir.

Nesnelerin ‘değişkenler’ olarak adlandırıldığını da görebilirsiniz. Programlama terimlerinde ikisi arasında fark vardır, ancak çok sık eş anlamlı olarak kullanılırlar.

Altıncı Nesneler oluşturma

- Aşağıdaki kodu kopyalayıp konsola yapıştırın.
- Eni 4 cm, boyu 10 cm bir dikdörtgenin alanı hesaplayınız.

```
# en nesnesi tanımlama  
  
# boy nesnesi tanımlama  
  
# alan nesnesi tanımlama  
  
# alan nesnesini yazdırma
```

[1] 40

- Eni 4 cm, boyu 10 cm bir dikdörtgenin köşegen uzunluğunu hesaplayınız.

```
# en nesnesi tanımlama  
  
# boy nesnesi tanımlama  
  
# köşegen nesnesi tanımlama  
  
# köşegen nesnesini yazdırma
```

[1] 10.77033

1.12.1 Ödev

DataCamp hesaplarınızda yer alan datacamp'tan size atanan bölümü tamamlayınız ve kitabın ilk bölümünü tamamlayınız.

Chapter 2

R Paketler

- R'yi yüklediğinizde, veri ileme ve istatistiksel analiz seçenekleri de dahil olmak üzere bir dizi fonksiyona erişebilirsiniz. Varsayılan kurulumda yer alan fonksiyonlar genellikle **Temel R/Base R** olarak adlandırılır ve birçok Temel R fonksiyonunu gösteren faydalı bir cheatsheet sayfası vardır [cheatsheet](#)
- **Temel R** telefonunuzda gelen varsayılan uygulamalar, paketleri ise ayrıca indirmeniz gereken ek uygulamalar olarak düşünmek faydalı olabilir.
- R fonksiyonları **ayrı paketler** halinde düzenlenmiştir. Böylece gerekli paketlerle çalışarak daha az bellek kullanımı ve hızlı işlem gücü sağlanır.
- Bu paketlerin bir başka avantajı da yazılan fonksiyonlardan oluşan paketlerin CRAN'den temin edilerek yüklenebilmesidir.
- Her paketin bir yaratıcısı ve kendisine ait bir yardım dosyası bulunur.

```
# paket yükleme
install.packages("CTT")
# paket aktive etme
library(CTT)
```

- Paket yükleme işlemi Rstudio'da yer alan menüler aracılığıyla da yapılabilir.
- R paketleri R fonksiyonlarının, verilerinin ve iyi derlenmiş bir formatta kodların kombinasyonlarından oluşmaktadır. `library()` komutu ile kişisel kütüphanenizdeki yüklü paketleri görebilirsiniz.
- Sadece temel pakette 1000'den fazla fonksiyon bulunmaktadır.

```
# temel paket fonksiyonlarına ulaşmak için
fonksiyonlar <- builtins()
length(fonksiyonlar)
```

```
## [1] 1380
```

```
fonksiyonlar[910:920]
```

```
## [1] "Cstack_info"          "crossprod"
## [3] "cospi"                "cosh"
## [5] "cos"                  "contributors"
## [7] "Conj"                 "conflicts"
## [9] "conflictRules"        "conditionMessage.condition"
## [11] "conditionMessage"
```

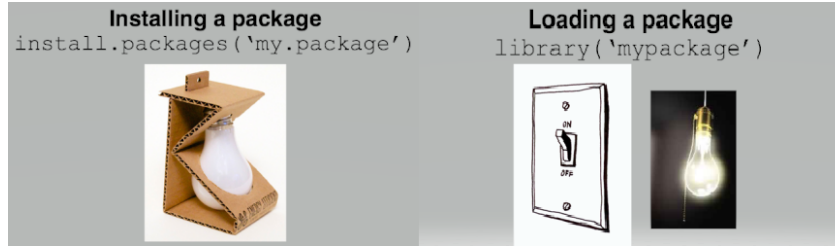


Figure 2.1: yüklet-etkinleştir

2.0.1 Altrma : tidyverse yükleme

- Bir paketi kullanabilmek için önce onu yüklemeniz gerekir. Aadaki kod, bu derste çok sık kullanacağımız bir paket olan **tidyverse** paketini yükler.

```
install.packages("tidyverse")
```

- Bir paketi yalnızca bir kez yüklemeniz gerekir, ancak R'yi her balattnzda kullanmak istediiniz paketleri yüklemeniz gerekir, benzer ekilde telefonunuza bir uygulamayı bir kez yüklemeniz gerekir, ancak her kullanmak istediinizde açmanız gerekir.

UYARI: WARNING: Rtools is required to build R packages” gibi bir hata mesaj alırsanız, [Rtools] (<https://cran.r-project.org/bin/windows/Rtools/>) adlı ekstra bir yazılım indirmeniz ve yüklemeniz gerekebilir.

2.0.2 Altrma : tidyverse etkinleştir

- Tidyverse'i etkinleştirmek için aadaki kodu çalıştırın.

```
library(tidyverse)
```

- Bir hata mesajı gibi görünen bir şey alacaksınız - öyle değil. Bu sadece R'nin size ne yaptığını anlatmasıdır.
- şimdi **tidyverse** paketini etkinleştirdiğimize göre, içerdiği fonksiyonlardan herhangi birini kullanabiliriz, ancak unutmayın, R'yi her balattnzda **library()** fonksiyonunu çalıştırmamız gerekir.

2.1 Github paketleri yükleme

- Bazı R paketleri github üzerinden yayınlanmaktadır. Bu paketleri standart yollarla yükleyemeyiz. Bu paketleri yüklemek için ilk olarak devtools paketinin yüklü olması gerekir.
- Bu paketlere bir örnek yapsal etkileşim modelleri ile ilgili APA formatında tablolar üreten semtools verilebilir. Paketin github sayfası linkte yer almaktadır. Paketin yüklenmesi için örnek kod aşağıda verilmiştir.

```
devtools::install_github("dr-JT/semoutput")
```

2.2 Yardım Sayfaları

- R'da temel ve diğer paketlerde yer alan fonksiyonların ilemleri görmek için yardım sayfalarını inceleyebilirsiniz. **? ve help()** fonksiyonları aynı işlevi sahiptir.

```
?is.na
```

```
help(sqrt)
```

- Örnein CTT paketini hem yüklediniz hem de etkinletirdiniz. Paket fonksiyon ve veri içeriini aadaki komutlarla görebilirsiniz.

```
# install.packages(CTT)
library(CTT)
ls("package:CTT")
data(package = "CTT") # yeni bir sekmede acilir.
?reliability
```

- Etkinletirdiiniz paketlerde yer alan fonksiyonların yardım sayfalarına ulaşabilirsiniz.

2.3 Paket çakmlar

- Daha da fazla fonksiyona sahip binlerce farklı R paketi vardır. Ne yazık ki, bazen farklı paketler aynı fonksiyon isimlerine sahiptir. Örnein, `dplyr` ve `MASS` paketlerinin her ikisi de `select()` adında bir fonksiyona sahiptir. Bu paketlerin her ikisini de yüklerseniz, R size bir çakma olduğunu söyleyen bir uyarı üretecektir.

```
library(dplyr)
library(MASS)
```

```
##
## Attaching package: 'MASS'
## The following object is masked from 'package:dplyr':
##
##      select
```

- Bu durumda, R size `dplyr` paketindeki `select()` fonksiyonunun aynı isimli başka bir fonksiyon tarafından gizlendiğini (veya ‘maskelendiğini’) söylüyor. Eğer `select()` fonksiyonunu kullanmay deneseydiniz, R en son yüklenen paketteki fonksiyonu kullanacaktı - bu durumda `MASS` fonksiyonunu kullanacaktı.
- Belirli bir fonksiyon için hangi paketi kullanmak istediğinizi belirtmek istiyorsanız, örnein `package::function` biçiminde kod kullanabilirsiniz:

```
dplyr::select()
MASS::select()
```

2.4 Paket Güncelleme

- R ve R Studio güncellemelerine ek olarak, paketlerin yazarları da bazen kodların günceller. Bu, bir pakete fonksiyon eklemek için olabileceği gibi hataları düzeltmek için de olabilir. **Kaçınılması gereken bir şey, yüklü bir paketi istemeden güncellemektir.**
- `install.packages()` fonksiyonunu çalıştırdığınızda, her zaman paketin en son sürümü yüklenir ve yüklemi olabileceğiniz eski sürümlerin üzerine yazılır. Bazen bu bir sorun değil, ancak bazen paket önemli ölçüde değişti için güncelleme kodunuzun artık çalışmadığını anlamına geldiğini görürsünüz. Bir paketin eski bir sürümüne geri dönmek mümkündür ancak yine de bundan kaçınmaya çalışın.

Bir paketin üzerine yanlışlıkla daha sonraki bir sürümün yazılması önlemek için, sizin veya bir başkasının kodu yanlışlıkla kaldırması ihtimaline karşı analiz komut dosyalarınıza `install.packages()` i asla dahil etmemelisiniz.

2.5 R ve RStudio'ya nasıl alıntı yapılır

- R'a atfıta bulunmanız ve referans vermeniz gereken bilimsel bir rapor yazmaktan biraz uzak olabilirsiniz, ancak zaman geldiğinde bunu onu getiren insanlara (çok ücretsiz!) kredi vermek için yapmak önemlidir. R, RStudio ve kullandığınız paketler için ayrı alıntılar salamalsınız.
- Kullandığınız R sürümü için atf almak için, size her zaman en son atf salayacak olan `citation()` fonksiyonunu çağırmanız yeterlidir.

```
citation()
```

```
##
## To cite R in publications use:
##
##   R Core Team (2022). R: A language and environment for statistical
##   computing. R Foundation for Statistical Computing, Vienna, Austria.
##   URL https://www.R-project.org/.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {R: A Language and Environment for Statistical Computing},
##     author = {{R Core Team}},
##     organization = {R Foundation for Statistical Computing},
##     address = {Vienna, Austria},
##     year = {2022},
##     url = {https://www.R-project.org/},
##   }
##
## We have invested a lot of time and effort in creating R, please cite it
## when using it for data analysis. See also 'citation("pkgname")' for
## citing R packages.
```

- Kullandığınız herhangi bir paket için atf oluşturmak için, atf yapmak istediğiniz paketin adıyla birlikte `citation()` ilevini de kullanabilirsiniz.

```
citation("tidyverse")
```

```
##
## To cite package 'tidyverse' in publications use:
##
##   Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R,
##   Golemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller
##   E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V,
##   Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to
##   the tidyverse." Journal of Open Source Software, 4(43), 1686.
##   doi:10.21105/joss.01686 <https://doi.org/10.21105/joss.01686>.
##
## A BibTeX entry for LaTeX users is
##
##   @Article{,
##     title = {Welcome to the {tidyverse}},
##     author = {Hadley Wickham and Mara Averick and Jennifer Bryan and Winston Chang and Lucy D'Agostini
##     year = {2019},
##     journal = {Journal of Open Source Software},
##     volume = {4},
##     number = {43},
```



```
##      pages = {1686},  
##      doi = {10.21105/joss.01686},  
##    }
```

- Kullandığınız RStudio sürümüne ait alıntı oluşturmak için `RStudio.Version()` fonksiyonunu kullanabilirsiniz:

```
RStudio.Version()
```

- Son olarak, yöntem bölümünüzün yazımında bunun nasıl görünebileceğine dair bir örnek:
Analiz R (R Core Team, 2020), RStudio (Rstudio Team, 2020) ve tidyverse paketi (Wickham, 2017) kullanılarak gerçekleştirilmiştir.
- Belirtildiği gibi, bunu bir süre yapmak zorunda kalmayabilirsiniz, ancak yaptığınızda buna geri dönmek çünkü açık kaynak topluluğuna çalışmalar için kredi vermek önemlidir.

Chapter 3

Fonksiyonlar

- Fonksiyon belli bir görevi yerine getirmek için yazılan bir grup komuttur.
- Fonksiyonların çalışması için girdilerinin olması gerekmektedir. Fonksiyonlar girdileri ile yaptıkları işlem sonucunda bir çıktı olutururlar.
- Fonksiyonlar girdileri o fonksiyonun çalışması için önceden belirlenen **argümanlar** ve o argümanların değerlerinden oluşur. (dilbilimle ilgileniyorsanız, bunlar bir özne ve nesne gerektiren fiiller olarak düşünmek isteyebilirsiniz)
- Fonksiyonların kullanımında üç noktaya dikkat edilmelidir.
 1. argümanların sırası
 2. argümanların olaan (default) değerleri
 3. bazı argümanların zorunlu, bazı argümanların opsiyonel olmasıdır
- Bir fonksiyonun aldığı tüm argümanlara yardım dokümantasyonunu kullanarak `?function` formatını kullanarak bakabilirsiniz. Bazı argümanlar zorunlu, bazıları ise isteğe bağlıdır. İsteğe bağlı olanlar, herhangi bir değer girmezseniz genellikle varsayılan/olaan (normalde yardım belgelerinde belirtilen) bir değer kullanır.
- Örnek olarak, normal dağılıma sahip bir sayı kümesini rastgele üreten `rnorm()` fonksiyonunun yardım belgelerine bakalım.
- Bir fonksiyonun aldığı tüm argümanlara yardım dokümantasyonunu kullanarak `?function` formatını kullanarak bakabilirsiniz. Bazı argümanlar zorunlu, bazıları ise isteğe bağlıdır. İsteğe bağlı olanlar, herhangi bir değer girmezseniz genellikle varsayılan/olaan (normalde yardım belgelerinde belirtilen) bir değer kullanır.

Alıştırma

- R Studio'yu açın ve konsola aşağıdaki kodu yazın:

```
?rnorm
```

- `rnorm()` için yardım belgeleri sağ alt yardım panelinde görünmelidir. Kullanım bölümünde, `rnorm()`un aşağıdaki formu alındığını görüyoruz:

```
rnorm(n, mean = 0, sd = 1)
```

- Argümanlar bölümünde, her bir argüman için açıklamalar bulunmaktadır. `n` oluşturmak istediğimiz gözlem sayısı, `mean` oluşturacağımız veri noktalarının ortalaması ve `sd` verinin standart sapmasıdır. Ayrıntılar bölümünde, `mean` ve `sd` için herhangi bir değer girilmezse, bu değerler için varsayılan olarak 0 ve 1 kullanılabileceği belirtilir. `n` için varsayılan bir değer olmadıysa, belirtilmesi gerekir, aksi takdirde kod çalışmaz.

- Bir örnek deneyelim ve R'den 5 rastgele say üretmesini istemek için gerekli `n` argümanın deitirelim.

Altrma II

- Aadaki kodu kopyalayp konsola yaptrn.

```
set.seed(12042016)
rnorm(n = 5)
```

```
## [1] -0.2896163 -0.6428964  0.5829221 -0.3286728 -0.5110101
```

- Bu sayların ortalamas 0 ve SD'si 1'dir. imdi farkl bir say kümesi üretmek için ek argümanlar deitirebiliriz.

```
rnorm(n = 5, mean = 10, sd = 2)
```

```
## [1] 13.320853  9.377956 10.235461  9.811793 13.019102
```

- Bu kez R yine 5 rastgele say üretti, ancak imdi bu say kümesi belirtildi gibi 10 ortalama ve 2 sd deerine sahip. Bir fonksiyonun hangi argümanlar gerektirdiini anlamanza yardmc olmas için yardım belgelerini kullanmay her zaman unutmayn.

Eer internette kod örneklerine bakyorsanz, sk sk `set.seed()` fonksiyonu ile balayan kodlar görebilirsiniz. Bu fonksiyon rastgele say üreticini kontrol eder - rastgele say üreten herhangi bir fonksiyon kullanıyorsanız (`rnorm()` gibi), `set.seed()` fonksiyonunu çaltrmak ayn sonucu almanız salayacaktır (baz durumlarda yapmak istediiniz ey bu olmayabilir). Bu örnekte `set.seed()` diyoruz, bu ayn rastgele sayılar alacağınız anlamna geliyor.

3.1 Argüman isimleri

- Yukardaki örneklerde, kodumuzdaki bamsz deiken adların yazdk (örnein, `n`, `mean`, `sd`), ancak bu kesinlikle gerekli deildir. Aadaki iki kod satrının her ikisi de ayn sonucu üretecektir (`rnorm()` fonksiyonunu her çaltrdınızda rastgele oldu için biraz farkl bir say kümesi üretecektir, ancak yine de ayn ortalama ve SD'ye sahip olacaklardır):

```
rnorm(n = 6, mean = 3, sd = 1)
rnorm(6, 3, 1)
```

- Önemli olarak, eer argüman isimlerini yazmazsanz, R argümanların varsayılan sırasn kullanacaktır, yani `rnorm` için girdiğiniz ilk sayının `n` olduunu varsayacaktır. ikinci say `mean` ve üçüncü say `sd`dir.
- Eer argüman isimlerini yazarsanz, argümanlar istediiniz sırada yazabilirsiniz:

```
rnorm(sd = 1, n = 6, mean = 3)
```

- R'yi ilk örenirken, fonksiyonun her bir parçasının ne yaptn hatırlamanza ve anlamanza yardmc olabileceinden, argüman adların yazmay yararlı bulabilirsiniz. Ancak, becerileriniz ilerledikçe argüman adların atlamay daha hızlı bulabilirsiniz ve ayrıca argüman adların kullanmayan çevrimiçi kod örnekleri göreceksiniz, bu nedenle her bir kod parçasının hangi argümana atfta bulunduunu anlayabilmek önemlidir (veya kontrol etmek için yardım belgelerine bakın).
- Bu derste, her bir fonksiyonu ilk kez kullandımızda argüman adların her zaman yazacaz, ancak sonraki kullanımlarda bunlar atlanabilir.

3.2 TAB ile otomatik tamamlama

- R Studio'nun çok kullanlı bir özelli, fonksiyonlar için TAB otomatik tamamlama özelliidir (bkz. ekil ??). Fonksiyonun adn yazp tab tuuna basarsanz, R Studio size fonksiyonun ald argümanlar ksa bir

açıklama ile birlikte gösterecektir. Argüman adnn üzerinde enter tuuna basarsanz, tşk telefonunuzdaki otomatik tamamlama gibi ad sizin için dolduracaktır. Bu, R’yi ilk öğrenirken inanılmaz derecede kullanılır ve bu özelliği sık sık kullanmayı **unutmamalısınız**.

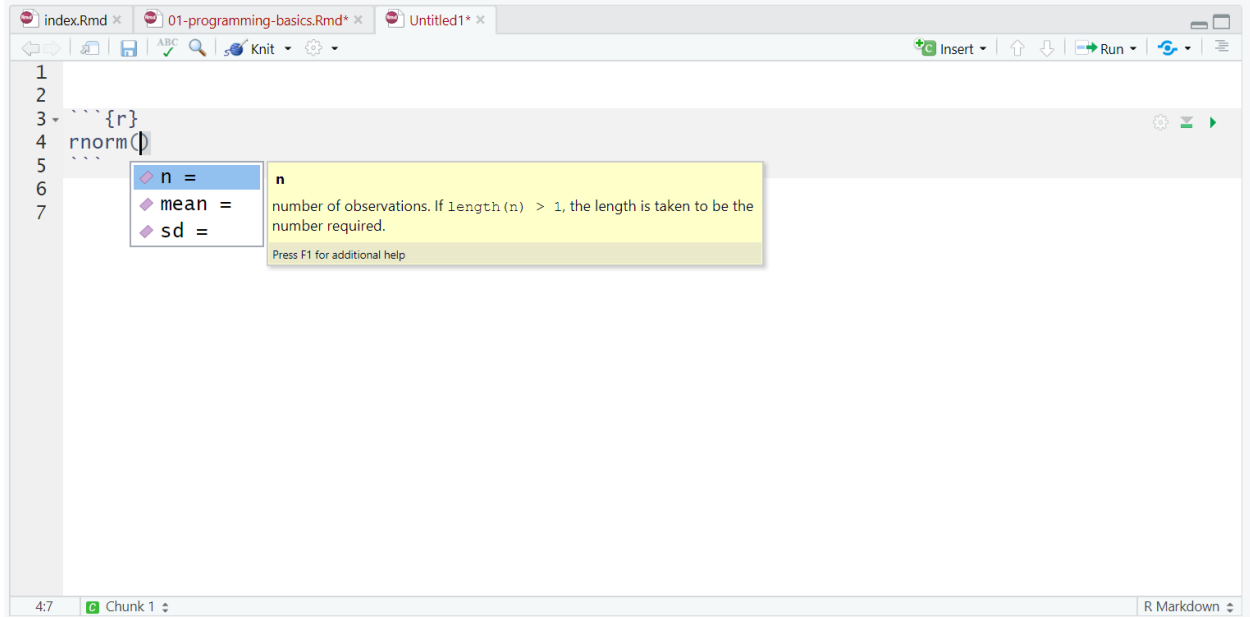


Figure 3.1: Tab ile otomatik durdurma

3.3 Kiisel tanıml fonksiyon

- Kiisel tanıml fonksiyon yazılmas aablonu aadaki gibidir.

```
fonksiyonadi <- function(argumanlar ve olagan degerleri){
  kodlar
  return()
}
```

- Oluturulan fonksiyon çalıtrırken ise aadaki ekinde çalıtrılır.

```
fonksiyonadi(argumanlar ve degerleri)
```

- Kare alma ilemi aadaki ekinde yapılabilir.

```
sayi <- 4
sayi * sayi
sayi ^2
```

```
## [1] 16
## [1] 16
```

- Bu işlem sürekli yapılacaksa fonksiyon olarak yazılabilir.

```
# kare alma fonksiyonu
kare_al <- function(sayi){
  return(sayi*sayi)
}
kare_al(4)
```

```
## [1] 16
```

- Farklı dereceden üsler alabilen bir fonksiyon yazalım.

```
#üs alma
üs_al<- function(x,us){
  return(x^us)
}
üs_al(3,4)
```

```
## [1] 81
```

- Argümanlardan birine olaan değer girilmesi

```
#üs alma
üs_al<- function(x,us=2){
  return(x^us)
}
üs_al(3) # us argumanın olagan değeri olan
# 2 oldu için argumana
# değeri girilmediğinde kare alır.
```

```
## [1] 9
```

- Aadaki fonksiyona 3 ve 4 değerleri girilirse çıktı ne olur?

```
myfunc <- function(x,y)
{
  a <- x+y
  b <- x*y
  return(a*b)
}
myfunc(3,4)
```

- `mean()` fonksiyonu en sık kullandığımız fonksiyonlardan biridir.

```
x <- c(1,2,3)
mean(x)
```

```
## [1] 2
```

- R base pakette yer alan bu fonksiyonu kendiniz de yazabilirsiniz.
- R’da deneyim kazandıkça, yaptığımız işlemler karmaşıklaştıkça kendi fonksiyonlarımız yazma ihtiyacı duyacaksınız.
- `avg()` isminde vektör ortalaması hesaplayan fonksiyon yazınız.
- Yazdığımız fonksiyon ile aadaki işlemi yapınız.

```
x <- 1:1000
avg(x)
```

```
## [1] 500.5
```

- Yazdığımız fonksiyon temel pakette yer alan `mean()` fonksiyonu ile aynı sonucu verdi mi?

```
identical(avg(x),mean(x))
```

```
## [1] TRUE
```

- Fonksiyon içinde tanımlanan nesneler çalışma alanına kaydedilmezler.
- Fonksiyonlar da R nesnesidir.

```
ls()
```

```
## [1] "avg"           "backtick"      "h1"
## [4] "kare_al"       "path"          "pkg"
## [7] "psyteachr_colors" "psyteachr_colours" "sayi"
## [10] "üs_al"        "x"
```

3.4 R Çalma Alan

- çalma alan, nesnelerin ve bilgilerin kaydedildiği alandır.
 - `ls()` ve `objects()` fonksiyonlar çalma alanında kayıtlı nesneleri konsolda göstermektedir.
 - `ls()` fonksiyonu ile nesneleri çarma ilemi özelleştirilebilir.
 - `ls.str()` fonksiyonu ise hafızadaki nesneleri ayrıntılar ile göstermektedir.
 - Çok fazla kod yazıyorsanız, environment (veya çalma alanı) birçok nesne ile darmadan olduğunu fark edebilirsiniz. Bu, hangi nesneye ihtiyacınız olduğunu bulmanız zorlaştırabilir ve bu nedenle yanlış veri seti kullanma riskiyle karşı karşıya kalabilirsiniz. Yeni bir veri kümesi üzerinde çalışıyorsanız veya son sürümünü elde etmeden önce çok sayıda farklı kod denediyseniz, yanlış nesneyi kullanmaktan kaçınmak için ortam/çalma alanı temizlemeyi unutmamak iyi bir uygulamadır. Bunu birkaç eklede yapabilirsiniz.
1. Nesneleri tek tek kaldırmak için konsola `rm(nesne_ad)` yazabilirsiniz. Önceki bölümde oluşturduğunuz nesnelerden birini kaldırmak için bunu şimdi deneyin.
 2. Ortamdaki tüm nesneleri temizlemek için konsolda `rm(list = ls())` komutunu çalıştırın.
 3. Ortamdaki tüm nesneleri temizlemek için ortam bölmesindeki süpürge simgesine de tıklayabilirsiniz.
 4. Konsolda yer alan işlemleri silmek için ise: CTRL + L (clear console) ya da süpürge iareti kullanılabilir.

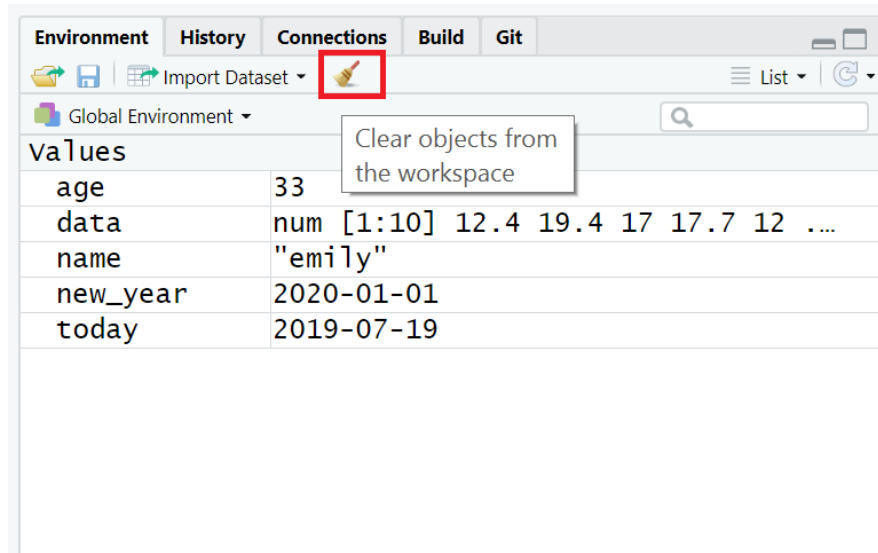


Figure 3.2: Clearing the workspace

3.5 R Çalma Dizini

- R yazılım Start/Başlangıç menüsü üzerinden çalıştırıldığında çalma dizini `C:/Users//Documents`
- Çalma dizinini sorgulamak için kullanılacak olan fonksiyon
 - `getwd()` (get working directory)

- Çalma dizinini deitirmek için kullanacak olan fonksiyon
 - `setwd()` (set working directory)
- Bu ilem Rstudio menuu “**Session**” sekmesinden ya da **CTRL + Shift + H** tular ile de yaplabilmektedir.

3.6 R’i Kapatma

- Kaydet (Save) ya da **CTRL + S** dosyadi.R uzantsyla kaydedilebilmektedir.
- Bu sayede tekrar kullanlabilmekte ya da bakalar ile kolaylkle paylalabilmektedir.
- Tüm programlar gibi “**x**” iareti ile ya da **q()** fonksiyonunu ile sonlandrlabilir.
- R’dan çık yaparken, program çalma alanının kaydedilip kaydedilmeyeceini sormaktadr.
- Eger R’in çalma alanı kaydetmesini istenirse, R çalma dizinine ‘Rdata uzantlı bir dosya kaydeder.
- Çalma alan kayd için `save.image("dosyaadi")` komutu da kullanlabilmektedir.
- R’dan çık yapmadan yapılan ilem durdurulmak istenirse, konsol bölümündeki “**Stop**” iareti veya **Esc** tular kullanlabılır.

3.7 R oturumlar

- R’yi açp kod yazmaya, paketleri yüklemeye ve nesneler oluturmaya baladınızda, bunu yeni bir **oturumda** yaparsınız. Çalma alanı temizlemeye ek olarak, bazen yeni bir oturum balatmak yararlı olabilir. Bu, bilgisayarınızda R’yi her balattınızda otomatik olarak gerçekleştir, ancak oturumlar sunucuda kalabilir. Kodunuzun çalmadn fark ederseniz ve nedenini bulamazsanız, yeni bir oturum balatmaya deer olabilir. Bu, ortamı temizleyecek ve yüklü tüm paketleri ayıraraktr - bunu telefonunuzu yeniden balatmak gibi düşünün.

3.8 Altrma

’Oturum - R’yi Yeniden Balat’a tıklayın.

3.9 Hata ayıklama ipuçları

- Kodlamann büyük bir kısmı kodunuzun neden çalmadn anlamaya çalmaktr ve bu acemi ya da uzman olmanız fark etmeksizin geçerlidir.
- Bu ders boyunca ilerlerken yaptığınız hataları ve bunları nasıl düzelttiğinizin kaydını tutmalısınız.
- Her bölümde dikkat etmeniz gereken bir dizi yaygın hata sunacağız, ancak üphesiz kendiniz de yeni hatalar yapacaksınız (ve düzelteceksiniz!).
- Kullanmaya çalıştığınız fonksiyonlar için doğru paketleri yüklediniz mi? Çok yaygın bir hata, paketi yüklemek için kodu yazmaktr, örneğin `library(tidyverse)` ancak daha sonra çalıştırmayı unutmaktır.
- Bir yazım hatası mı yaptınız? Unutmayın `data` ile `DATA` aynı şey değildir ve `t.test` ile `t_test` aynı şey değildir.
- Bir paket çıkmıyor mu var? Paket ve fonksiyonu `package::function` ile belirtmeyi denediniz mi?
- Bu kesinlikle bir hata mı? R’deki tüm kırmızı metinler hata anlamına gelmez - bazen size sadece bilgi içeren bir mesaj verir.

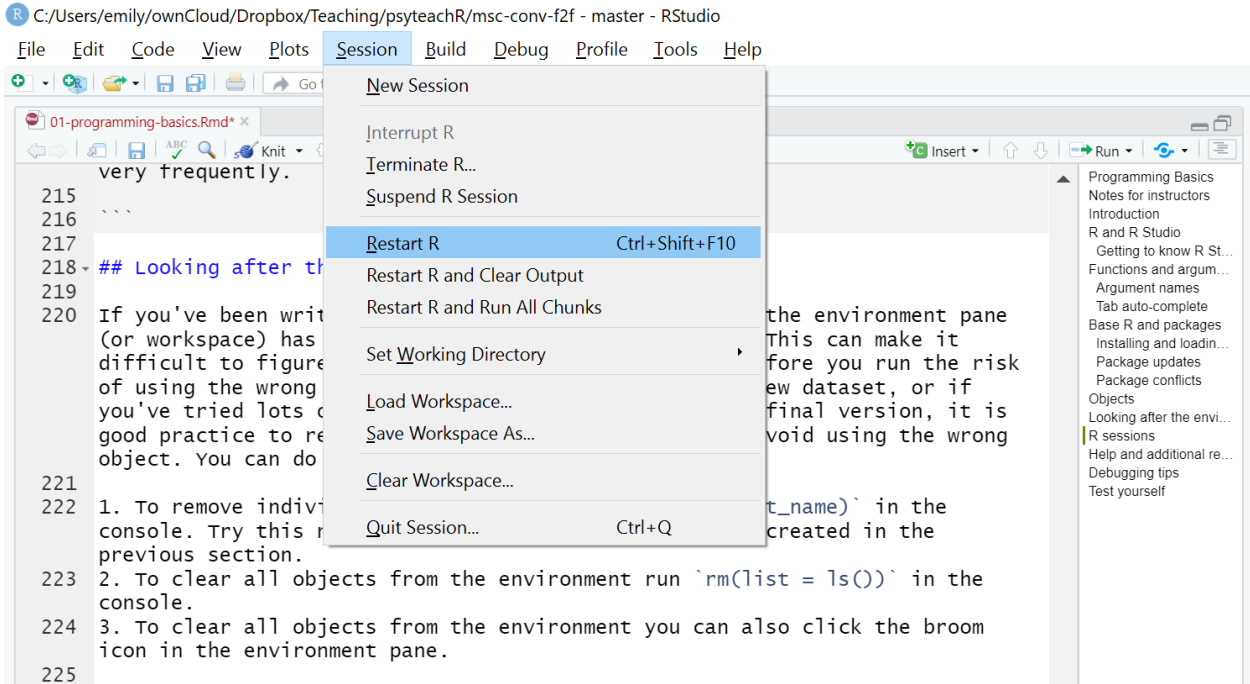


Figure 3.3: The truth about programming

3.10 Yardmc Kaynaklar

Programlamada iyi olmak demek, bir eyler denemek, internette yardm aramak ve kopyalanacak kod örnekleri bulmak demektir.

- etkili bir ekilde problem çözmeyi öğrenmek, bu kurs boyunca gelitirmeniz gereken temel bir beceridir.
- Yardm belgelerini kullann. Bir fonksiyonun nasl çaltın anlamakta zorlanyorsanz, `?function` komutunu hatrlayn.
- Bir hata mesaj alrsanz, kopyalayp Google'a yaptrn - büyük olaslıkla baka biri de ayn sorunu yaamtr.
- Bu ders materyallerine ek olarak, R öğrenmek için bir dizi mükemmel kaynak vardır:
 - R Cookbook
 - StackOverflow
 - Veri Bilimi için R
 - Twitter'da `#rstats` hashtag'ini arayn veya kullann

3.11 Altrma : Kendini test et

Soru 1. Neden `install.packages()` kodunu analiz kodlarında asla dahil **etmemelisiniz**?

- (A) Bunun yerine `library()` kullanmalsnz
- (B) Paketler zaten temel R'n bir parçasdr
- (C) Siz (veya bir bakas) yanllkla kodunuzun çalması durduran bir paket güncellemesi yükleyebilirsiniz
- (D) Paketin en son sürümüne zaten sahipsiniz

Açıklama

Unutmayın, `install.packages()` ilevini çalıştırdığınızda her zaman paketin en son sürümü yüklenir ve yüklemi olabileceğiniz eski sürümlerin üzerine yazılır.

Soru 2. Aadaki kod ne üretecektir?

```
rnorm(6, 50, 10)
```

- (A) Ortalaması 6 ve SD'si 50 olan 10 sayıdan oluşan bir veri seti
- (B) Ortalaması 50 ve SD'si 10 olan 6 sayıdan oluşan bir veri seti
- (C) Ortalaması 10 ve SD'si 6 olan 50 sayıdan oluşan bir veri seti
- (D) Ortalaması 10 ve SD'si 6 olan 50 sayıdan oluşan bir veri seti

Açıklama

`rnorm()` için varsayılan biçim `rnorm(n, mean, sd)` şeklindedir. Bir fonksiyonun her bir argümanının ne ile yaradın hatırlamak için yardım ihtiyacınız varsa, `?rnorm` komutunu çalıştırarak yardım belgelerine bakın

Soru 3. Aynı isimde fonksiyonlara sahip iki paketiniz varsa ve tam olarak hangi paketin kullanılacağını belirtmek istiyorsanız, hangi kodu kullanırsınız?

- (A) `package::function`
- (B) `function::package`
- (C) `library(package)`
- (D) `install.packages(package)`

Açıklama

Örneğin `dplyr::select` gibi `package::function` biçimini kullanmalısınız. Paketlerinizi ilk yüklediğinizde, herhangi bir fonksiyon aynı isme sahipse R'nin sizi uyaracağı unutmayın - buna dikkat etmeyi unutmayın!

Soru 4. Aadakilerden hangisinin bir argüman olması en muhtemeldir?

- (A) 35
- (B) `read_csv()`
- (C) `<-`

Soru 5. Fonksiyonlar oluşturmanın için aadakilerden hangisini kullanmalısınız?

- (A) `()`
- (B) `[]`
- (C) `{}`

.

Soru 6. `<-`'nin görevi, fonksiyondan elde edilen çıktıyı bir/bir atamaktır.

- (A) nesne
- (B) atama
- (C) arguman

Sra Markdownda

Codeacedemy

3.12 Ödev

- Sadece temel pakette 1500'e yakın fonksiyon bulunduu için ders d altrmalar yapmanız gereklidir.
- R kurulumu ile ilgili learnr paketi hazırlanmış bir interaktif altrma örneğini inceleyiniz.
- Kitap Bölüm 1 altrmalarını tamamlayınız.
- Datacamp da üzerine atanan bölüm altrmalarını tamamlayınız.
- swirl package **learn R in R** paketi yükleyerek altrma yapmayı deneyiniz.
- Referans kart sayfasının çıktısını alarak duvarınıza asmanız öneririm.

Chapter 4

Vektörler

- R lineer cebir temelli bir programlama dilidir.
- Vektörler tek boyutludur.
- R'da vektörler birletirmek (combine/concatenate) anlamına gelen `c()` fonksiyonu ile oluşturulmaktadır.
- R'da veriler bir araya gelerek veri yapıları oluşturulur.
 - vektör (vector)
 - liste (list)
 - matris (matrix)
 - veri seti (data.frame)
 - dizi (array)

4.1 Vektör Oluşturma

```
(sayisal_vektor <- c(1,2,3))  
  
## [1] 1 2 3  
(karakter_vektor <- c("a","b","c")) ## çift tırnak  
  
## [1] "a" "b" "c"  
(mantıksal_vektor <- c(TRUE,TRUE,FALSE))  
  
## [1] TRUE TRUE FALSE
```

4.2 Vektör lemleri

- Vektör uzunluğu `length()` fonksiyonu ile vektör türleri ise `class()`, `mode()` ya da `typeof()` fonksiyonları ile tür belirlemek için kullanılmaktadır.
- Vektörler bir veya daha fazla elemandan oluşabilmektedir.

```
a <- 1 # tek elemandan oluşur.  
# Vektör uzunluğunu öğrenmek için length() fonksiyonu  
length(a)  
  
## [1] 1
```

```
x <- 1:10
```

- bir vektöründeki verilerin toplanması `sum(x)` 55
- bir vektöründeki verilerin çarpılması `prod(x)` 3.6288×10^6
- bir vektöründeki verilerin küçükten büyüye sıralanması `sort(x)` 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- bir vektörünün elemanların sıralarını tersine çevrilmesi `rev(x)` 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
- bir vektöründeki verilerin standart sapmasının hesaplanması `sd(x)` 3.0276504
- bir vektöründeki en büyük verinin gösterilmesi `max(x)` 10
- bir vektöründeki en küçük verinin gösterilmesi `min(x)` 1
- En büyük verinin vektörün kaçın eleman olduğunu gösterilmesi `which.max(x)` 10
- En küçük verinin vektörün kaçın eleman olduğunu gösterilmesi `which.min(x)` 1
- Vektörlerden eleman sırası, isim ve mantıksal operatörler olmak üzere üç farklı yolla eleman seçilebilir.

```
ad <- c("Ali", "Elif", "Su", "Deniz",  
"Aras", "Berk", "Can", "Ece", "Efe", "Arda")
```

- `ad` vektörünün 1. elemanı `ad[1]` Ali
- `ad` vektörünün 5. elemanın yazdıracak kodu oluturunuz. _____
- `ad` vektörünün son elemanın yazdıracak kodu oluturunuz. _____
- `ad` vektörünün son elemanın yazdıracak kodu vektörün 10 elemanlı olduğunu bilmediğiniz de ne yaparsınız?

- Vektörün sadece 1., 5. 8 elemanın yazdıracak kodu oluturunuz. _____
- Vektörün sadece 1. elemanın hariç tutacak kodu oluturunuz _____
- Vektörün 1. ve 5. elemanın hariç tutacak kodu oluturunuz _____
- Vektörün son üç elemanın yazdıracak kodu oluturunuz _____

4.3 Vektöre eleman eklenmesi

```
ad[11] <- "Asu"; ad
```

```
## [1] "Ali" "Elif" "Su" "Deniz" "Aras" "Berk" "Can" "Ece" "Efe"  
## [10] "Arda" "Asu"
```

- vektöre birden fazla eleman eklenmesi

```
ad[12:13] <- c("Ahu", "Han"); ad
```

```
## [1] "Ali" "Elif" "Su" "Deniz" "Aras" "Berk" "Can" "Ece" "Efe"  
## [10] "Arda" "Asu" "Ahu" "Han"
```

- Vektörün ortasına eleman eklenmesi `append()` fonksiyonu ile yapılabilir. Fonksiyon yardım sayfasını inceleyiniz.

```
(ad <- append(ad, "Taha", after = 3))
```

```
## [1] "Ali" "Elif" "Su" "Taha" "Deniz" "Aras" "Berk" "Can" "Ece"  
## [10] "Efe" "Arda" "Asu" "Ahu" "Han"
```

- ya da `c()` fonksiyonu ile yapılabilir.

```
ad <- c(ad[1:5], "Selim", ad[7:length(ad)]); ad
```

```
## [1] "Ali" "Elif" "Su" "Taha" "Deniz" "Selim" "Berk" "Can" "Ece"
## [10] "Efe" "Arda" "Asu" "Ahu" "Han"
```

4.4 Altrma

- 10 kiiden oluan bir gruptaki kiilerin boy ve kilo ölçümleri için ise aadaki vektör oluturulmutur.

```
ad <- c("Ali", "Elif", "Su", "Deniz",
"Aras", "Berk", "Can", "Ece", "Efe", "Arda")
boy <- c(160, 165, 170, 155, 167, 162, 169, 158, 160, 164)
kilo <- c(50, 55, 57, 50, 48, 65, 58, 62, 45, 47)
```

- Eer elimizdeki vektör isimlendirilmi bir vektör ise eleman seçimini isimle de yapabiliriz.

```
#isimsiz boy vektörü
names(boy) # names() fonksiyonu ile isimlendirme yapılabilir.
```

```
## NULL
```

- ad vektörünü boy vektörü ile isimlendirirken nasl kullanabiliriz? _____
- Arda 'nn boyunu isimlendirilmi vektörü kullanarak boy["Arda"] ile yazdrsrnz

4.5 Örüntülerle Vektör Oluturma

- Vektör oluturmann farkl yollar bulunmaktadr.
- En basit yolu iki nokta ":" operatörünü kullanmaktr.

```
rakamlar <- 0:9
rakamlar
```

```
## [1] 0 1 2 3 4 5 6 7 8 9
```

- büyükten küçüie rakamlardan vektör oluturulmas

```
rakamlar <- 9:0
rakamlar
```

```
## [1] 9 8 7 6 5 4 3 2 1 0
```

4.5.1 seq()

- Belirli bir kurala göre say dizileri oluturmak için ise seq(), rep() ve paste() fonksiyonlarından yararlanabilir. İlk olarak bu fonksiyonların yardım sayfalarını inceleyelim.
- 1'den 10'a kadar birer birer artan sayılardan dizi oluturulacak kodu oluturunuz. seq(from=1, to=10, by=...)

—

- Bir önceki ilemi argümanşz olarak oluturunuz. _____
- Aynı çkty tek bir argümanla elde edebilir misiniz? _____
- length argümanın kullanarak aadaki çkty oluturacak kodu oluturunuz. _____

```
## [1] 1.0 1.4 1.8 2.2 2.6 3.0
```

- by argümanın ile art miktarı kullanarak aadaki çkty oluturacak kodu oluturunuz. _____

```
## [1] 1.0 1.5 2.0 2.5 3.0
```

- Belirli bir aralıkta kaç elemanın yer alacağını `length.out` argümanı kullanarak aadaki çıktıyı oluturacak kodu oluturunuz. _____

```
## [1] 1.0 1.5 2.0 2.5 3.0
```

4.5.2 rep()

`rep()` fonksiyonu için örnekler

```
# üç elemanlı bir vektörün üç kere tekrar ettirilmesi
rep(c(3,4,5), 3)
```

```
## [1] 3 4 5 3 4 5 3 4 5
```

```
# rakamların üç kere tekrar ettirilmesi
rep(0:9, times = 3)
```

```
## [1] 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
```

- `a <- c(3,5,7)` vektörünü kullanarak aadaki çıktıyı elde edecek kodu hazırlaynz. _____

```
## [1] 3 3 3 5 5 5 7 7 7
```

- `a <- c(3,5,7)` vektörünü kullanarak aadaki çıktıyı elde edecek kodu hazırlaynz. _____

```
## [1] 3 3 3 5 5 5 7 7 7 3 3 3 5 5 5 7 7 7 3 3 3 5 5 5 7 7 7
```

- Çıktıyı elde edecek kodu hazırlaynz. _____

```
## [1] 1 1 2 2 3 3 4 4
```

- Çıktıyı elde edecek kodu hazırlaynz. _____

```
## [1] 1 2 2 3 3 3
```

4.5.3 paste()

- `paste()` fonksiyonu çkts her zaman için karakterdir.

```
paste(1:4) # çkts karakterdir
```

```
## [1] "1" "2" "3" "4"
```

```
class(paste(1:4))
```

```
## [1] "character"
```

- Çıktıyı elde edecek kodu tamamlaynz `paste("test",...)` _____

```
## [1] "test 1" "test 2" "test 3" "test 4" "test 5" "test 6" "test 7"
```

```
## [8] "test 8" "test 9" "test 10"
```

- Çıktıyı elde edecek kodu tamamlaynz `paste("test",1:10,"...",sep="_")` _____

```
## [1] "test_1_puan" "test_2_puan" "test_3_puan" "test_4_puan" "test_5_puan"
```

```
## [6] "test_6_puan" "test_7_puan" "test_8_puan" "test_9_puan" "test_10_puan"
```

- Çıktıyı elde edecek kodu tamamlaynz `paste("test",c("A","B","C","D",...))` _____

```
## [1] "test A" "test B" "test C" "test D" "test 1" "test 2" "test 3" "test 4"
```

4.6 Rasgele Veri Oluturma

- Farklı fonksiyonlarla rastgele veri üretilebilir. Örnek 0-100 arasında 20 farklı değer elde edilmek istenilsin. Bunu yapmak için `sample()`, `runif()` ya da `rnorm()` fonksiyonlarından yararlanılabilir.

```
sample(0:100,5)
```

```
## [1] 83 54 94 87 1
```

```
runif(10, 0, 5)
```

```
## [1] 3.5227613 4.0081999 3.7562672 2.3261348 2.5009933 2.0598770 4.1288741
```

```
## [8] 4.7385931 0.7225128 1.4613490
```

```
rnorm(10,50,5)
```

```
## [1] 42.88861 50.54615 47.05873 50.16599 46.81834 53.32444 43.01495 47.79592
```

```
## [9] 40.80267 41.49262
```

- Kullanılan üç fonksiyonun da yardım sayfaları ve kullanım amaçlarını inceleyiniz.

4.7 İlemler

BKI vücut ağırlığının metre cinsinden boy uzunluğunun karesine bölünmesi ile elde edilmektedir. Her bir bireye ait BKI değerini hesaplayınız. BKI değerlerinin ortalamasını kaçtır (iki ondalık yuvarlayınız)? _____

```
ad <- c("Ali","Elif","Su","Deniz","Aras","Berk","Can","Ece","Efe","Arda")
boy <- c(160,165,170,155,167,162,169,158,160,164)
kilo <- c(55,55,57,50,48,65,58,62,45,47)
```

Çözüm!, bakmadan yapmayın!

```
# BKI hesaplanması
```

```
boy_m <- boy/100
```

```
BKI <- kilo/( boy_m * boy_m)
```

```
BKI
```

```
round(mean(BKI),2)
```

```
## [1] 21.48437 20.20202 19.72318 20.81165 17.21109 24.76757 20.30741 24.83576
```

```
## [9] 17.57812 17.47472
```

```
## [1] 20.44
```

4.7.1 Kendinizi Test Edin

S1. Aadaki tabloda yer alan üç sütun için birer vektör oluturunuz. Öğrencilerin geçme notu her iki sınavın ortalaması olarak hesaplanacaktır. Bu öğrencilerin geçme notlarını hesaplayınız. Geçme notlarının betimsel istatistiklerini hesaplayınız.

Öğrenci	Vize	Final
Oğrenci1	50	45
Oğrenci2	55	65
Oğrenci3	60	85
Oğrenci4	70	90
Oğrenci5	80	85

Geçme notlarının minimum değeri: _____

Geçme notların ortalama değeri: _____

Geçme notların maksimum değeri: _____

S2. Birden n 'e kadar olan sayların toplamı hesaplayan fonksiyon yazın `toplam()` tek argümanlı fonksiyon oluturunuz. Argüman değeri 5 olduğunda $1+2+3+4+5=15$ değerini versin.

- birden n 'e kadar olan sayların toplamı: $(n*(n+1))/2$

S3. 1'den n 'e kadar olan sayların toplamı hesaplayan fonksiyonu argümanlı olarak aadaki şekilde yazmay deneyiniz. Fonksiyonu çalıştırdığınızda ekranda/konsolda kaç tane kadar olan sayların toplamı hesaplandı yazsın, kullanıcının girdi değeri göre aada çıktı çıksın.

```
toplam()
```

```
kaç tane kadar olan sayların toplamı hesaplandı: 10
```

```
[1] " 10 'e kadar olan sayların toplamı: 55"
```

4.8 ÖDEV

- Kitap Bölüm 2 1. Soruyu tamamlayınız.
- swirl package - learn R in R (Programming ilk 6 bölüm)
- datacamp ödevinizi yapmayı unutmayın

Chapter 5

R Nesneler

- Örnek bir veri seti

```
library(tidyverse)
data(diamonds)
head(diamonds)
```

```
## # A tibble: 6 x 10
##   carat cut          color clarity depth table price      x      y      z
##   <dbl> <ord>        <ord> <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23 Ideal      E      SI2     61.5   55   326   3.95   3.98   2.43
## 2  0.21 Premium   E      SI1     59.8   61   326   3.89   3.84   2.31
## 3  0.23 Good      E      VS1     56.9   65   327   4.05   4.07   2.31
## 4  0.29 Premium   I      VS2     62.4   58   334   4.2    4.23   2.63
## 5  0.31 Good      J      SI2     63.3   58   335   4.34   4.35   2.75
## 6  0.24 Very Good J      VVS2    62.8   57   336   3.94   3.96   2.48
```

R nesne (object) yönelimli bir programlama dilidir.

- Karakter (character)
- Sayısal (numeric)
 - tam sayı (integer)
 - ondalıklı sayı (double)
 - karmaşık sayı (complex)
- Mantıksal (logical)
- Faktör (factor)
- Liste (list)
- Fonksiyon (function)

5.1 tam sayı

- tamsayı nesnesi oluşturulması

```
tamsayi <- 2L
```

- tamsayı nesnesinin türünün sorgulanması

```
typeof(tamsayi)
```

```
## [1] "integer"
```

- tamsayı nesnesinin yazdırılması

```
tamsayi
```

```
## [1] 2
```

5.2 ondalk say

- ondaliksayi nesnesinin oluturulmas

```
ondaliksayi <- 2.5
```

- ondaliksayi nesnesinin türünün sorgulanmas

```
typeof(ondaliksayi)
```

```
## [1] "double"
```

- ondaliksayi nesnesinin yazdırılması

```
ondaliksayi
```

```
## [1] 2.5
```

5.3 lemler

- tek elemanlı vektörler

```
x <- 1
```

```
y <- 1
```

```
x+y
```

```
## [1] 2
```

- çok elemanlı vektörler

```
x <- c(3,4,5)
```

```
y <- c(1,2,3)
```

```
# vektör eleman sayılar aynı mı?
```

```
length(x)==length(y)
```

```
## [1] TRUE
```

```
x+y
```

```
## [1] 4 6 8
```

```
x-y
```

```
## [1] 2 2 2
```

- çok elemanlı vektörler

```
x <- 1:9
```

```
y <- c(1,2,3)
```

```
# vektör eleman sayılar farklı mı?
```

```
length(x)/length(y)
```

```
## [1] 3
```

```
x+y
```

```
## [1] 2 4 6 5 7 9 8 10 12
```

```
x/y
```

```
## [1] 1.0 1.0 1.0 4.0 2.5 2.0 7.0 4.0 3.0
```

- çok elemanlı vektörler

```
x <- 1:5
y <- c(1,2)
# vektör eleman sayılar farklı olduğunda
length(x)/length(y)
```

```
## [1] 2.5
```

- $x+y$ ileminin sonucu nedir? _____

Çözüm

```
x + y
```

```
## Warning in x + y: longer object length is not a multiple of shorter object
## length
## [1] 2 4 4 6 6
```

5.4 Karakter Nesneler

- karakter nesnesi oluşturulması

```
karakter <- "olcme"
```

- Oluşturulan nesnenin türünün sorgulanması

```
typeof(karakter)
```

```
## [1] "character"
```

- nesne yazdırılması

```
karakter
```

```
## [1] "olcme"
# karakter nesnesi oluşturulması
ad <- "Su"
soyad <- "Sevim"
```

- iki nesneyi arada boşluk bırakarak birleştirir.

```
paste(ad,soyad)
```

```
## [1] "Su Sevim"
```

- sep argümanı farklı eklerle özelleştirilebilir.

```
paste(ad,soyad, sep="_")
```

```
## [1] "SuSevim"
```

```
paste(ad,soyad,sep=" ")
```

```
## [1] "Su Sevim"
```

- base pakette yer alan bazı karakter vektörleri bulunmaktadır.

```
letters
```

```
## [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

```
LETTERS
```

```
## [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
## [20] "T" "U" "V" "W" "X" "Y" "Z"
```

```
month.name
```

```
## [1] "January" "February" "March" "April" "May" "June"
## [7] "July" "August" "September" "October" "November" "December"
```

```
month.abb
```

```
## [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
```

- Nesne birleştirme fonksiyonlarından en sık kullanan `paste()`
- `paste()` fonksiyonunun temel argümanları ise `sep` ve `collapse`'dir.

```
harf5 <- letters[1:5]
(harf51 <- paste(harf5, 1:5, sep="_"))
```

```
## [1] "a_1" "b_2" "c_3" "d_4" "e_5"
```

```
length(harf51)
```

```
## [1] 5
```

```
(harf52 <- paste(harf5, 1:5, sep="_",
collapse=" "))
```

```
## [1] "a_1 b_2 c_3 d_4 e_5"
```

```
length(harf52)
```

```
## [1] 1
```

- `paste()` fonksiyonunun yardım sayfasını inceleyiniz.

5.4.1 Günün Sorusu

- Aadaki çıktıyı oluturacak olan kodu oluturunuz. Madde güçlük değerlerini `runif()` fonksiyonu ile oluturabilirsiniz.

```
## [1] "1. maddenin guclugu: 0.92" "2. maddenin guclugu: 0.48"
## [3] "3. maddenin guclugu: 0.93" "4. maddenin guclugu: 0.39"
## [5] "5. maddenin guclugu: 0.34" "6. maddenin guclugu: 0.6"
## [7] "7. maddenin guclugu: 0.01" "8. maddenin guclugu: 0.34"
## [9] "9. maddenin guclugu: 0.53" "10. maddenin guclugu: 0.66"
```

Bunun birden fazla yolu olabilir, farklı eklerle yapabilirsiniz.

Büyük Küçük Harf Düzenleme Fonksiyonları `toupper()` ve `tolower()`

```
toupper(harf5)
```

```
## [1] "A" "B" "C" "D" "E"
```

```
tolower(harf5)
```

```
## [1] "a" "b" "c" "d" "e"
```

casefold() fonksiyonu da upper argüman ile birlikte kullanılabilir.

```
casefold(harf5, upper = FALSE)
```

```
## [1] "a" "b" "c" "d" "e"
```

```
casefold(harf5, upper = TRUE)
```

```
## [1] "A" "B" "C" "D" "E"
```

- Karakter nesnelerin kaç harften olutuu nchar() fonksiyonu ile belirlenebilir.

```
nchar(month.name)
```

```
## [1] 7 8 5 5 3 4 4 6 9 7 8 8
```

- Karakter nesneleri belli bir yerden bölmek için substr() ve substring() fonksiyonlar kullanılabilir.

```
substr("YILMAZ", 1,3)
```

```
## [1] "YIL"
```

- substring("YILMAZ", 1:.. , 1:6) kodunda “Y” “I” “L” “M” “A” “Z” çkts oluturacak kodu yaznz
—
- ‘substring("YILMAZ", ... , 4:6) kodunda “ILM” “ILMA” “ILMAZ”‘ çktns oluturacak kodu yaznz
—
- Karakter nesnelerde daha fazlas için aadaki fonksiyonlar inceleyebilirsiniz.
- strsplit()
- noquote()
- cat()
- grep()
- duplicated()
- agrep()

5.5 Mantksal Nesneler

```
mantiksal1 <-TRUE
```

```
typeof(mantiksal1)
```

```
## [1] "logical"
```

```
mantiksal1
```

```
## [1] TRUE
```

Mantksal operatörler programlamann temeli ve vazgeçilmezidir.

```
# eitlik snamas
```

```
T==TRUE
```

```
## [1] TRUE
```

- 4==5 kodunun sonucu nedir? _____
- 4<5 kodunun sonucu nedir? _____
- 10>100 kodunun sonucu nedir? _____

- Mantksal operatörlerle yapılan snamalar ile mantksal nesneler oluturulur.

```
sonuc <- 4<5
typeof(sonuc)
```

```
## [1] "logical"
```

- Nesne türleri arasındaki deim uygunluk durumuna göre `as.*()` fonksiyonlar ile salanr.

```
# Karakter veri numerik veriye
as.numeric("3.14")
```

```
## [1] 3.14
```

```
# ondalk verin tam sayya
as.integer(pi)
```

```
## [1] 3
```

```
# karakter veri numerik veriye (NA)
as.numeric("olcme")
```

```
## Warning: NAs introduced by coercion
```

```
## [1] NA
```

```
# mantksal veri karakter veriye (NA)
as.character(TRUE)
```

```
## [1] "TRUE"
```

```
# numerik veri karakter veriye
as.character(10)
```

```
## [1] "10"
```

```
# mantksal veri numerik veriye
as.numeric(TRUE)
```

```
## [1] 1
```

5.6 Nesne Türleri Sorgulama

- Nesne türleri sorgulamak için ise `class()` ya da `mode()` fonksiyonlar kullanabilir. Ancak bir nesne türüne ait olup olmadn sorgulamak için ise `is.*()` fonksiyonlar kullanlr.

```
x<- 3.14; class(x)
```

```
## [1] "numeric"
```

```
is.numeric(x)
```

```
## [1] TRUE
```

```
is.logical(x)
```

```
## [1] FALSE
```

- Saysal nesnelerin türü için `typeof()` fonksiyonu da kullanlabılır.

```
y <- 2L; typeof(y); class(y) # satir içi kod ayırma
```

```
## [1] "integer"
```

```
## [1] "integer"
```

```
is.integer(y)
```

```
## [1] TRUE
```

```
is.double(y)
```

```
## [1] FALSE
```

5.6.1 Günün Sorusu

- aada yer alan **ad_soyad** nesnesini kullanarak

```
ad_soyad<- c("Ayse-Sel","Can-Yucel","Cem-Togay","Banu-Cift")
```

aadaki çıktıyı oluturmaya çalışınız.

```
## [1] "Ayse" "Can"  "Cem"  "Banu"
```

```
## [1] "Sel"   "Yucel" "Togay" "Cift"
```

Chapter 6

Markdown

Bu bölümde kullanacağımız paketler unlardır.

```
knitr::opts_chunk$set(message = FALSE, warning = FALSE)
# bu bölüm için gerekli paketler
library(tidyverse) # çeitli veri manipülasyon fonksiyonlar
library(knitr) # tablo ve görüntü gösterimi için
library(kableExtra) # tablolar ekilemek için
library(papaja) # APA tarz tablolar için
library(gt) # süslü tablolar için
library(DT) # etkileimli tablolar için
```

ndir R Markdown Cheat Sheet.

R markdon sunusu

markdown hatırlatıcı notlar

6.1 Neden tekrarlanabilir raporlar kullanılmalı?

Bir rapor hazırladığınız düşünün

- içinde analiz sonuçları olan tablolar
- grafikler ve görsel olsun
- bu raporu güncellemeniz gerektiğinde veri seti, grafikler, analizler baka dizinlerde olabilir.
- Tekrarlanabilir raporlar, tüm analizleri gerçekleştirmek ve tablolar oluşturmak için gereken kodla birlikte rapor metnini tek bir belgede bir araya getirmektir.
- Bu, balangçta biraz fazladan çaba gerektirse de, herhangi bir değişiklik olduğunda tek bir dümeye basarak tüm raporunuzu güncellemenizi sağlayarak size fazlasıyla geri ödeme yapacaktır.
- Araştırmalar ayrıca, bilimsel literatürdeki makalelerin çoğunda olmasa bile birçoğunda raporlama hataları olduğunu göstermektedir. Tekrarlanabilir raporlar, transkripsiyon ve yuvarlama hatalarını önlemeye yardımcı olur.
- Rapor daha sonra orijinal formattan HTML, word ve ya PDF gibi daha tanılabilir baka bir formata “derlenir”. Bu, örneğin Microsoft Excel’de veya SPSS gibi bir istatistik programında bir grafik oluşturup

ardından bunu Microsoft Word’e yaptırınız geleneksel kesme ve yapıştırma yaklaşımlarından farklıdır.

6.2 Bir proje düzenlemek

- İlk olarak, organize olmamız gerekiyor. RStudio’daki projeler, bir proje için ihtiyaç duyduğunuz tüm dosyalar gruplandırmanın bir yoludur. Çoğu proje komut dosyaların, veri dosyaların ve komut dosyası veya görüntüler tarafından oluşturulan PDF raporu gibi çıktı dosyaların içerir.

6.2.1 Dosya Sistemi

- Bilgisayarınızın dosya sistemi, hem dosyalar hem de “alt dizinleri” içeren büyük dizin gibidir. Bir dosyanın konumunu adıyla ve içinde bulunduğu tüm dizinlerin adlarıyla belirtebilirsiniz.
- Örneğin, Kubra Masaüstünde `report.Rmd` adında bir dosya varsa, tam dosya yolunu u ekinde belirtebilir: `/Users/Kubra/Desktop/report.Rmd`, çünkü Masaüstü dizini, tüm dosya sisteminin tabanında bulunan Kullanıcılar/Users dizininin içindeki Kubra dizininin içindedir. Bu dosya masaüstünüzde olsaydı, kullanıcı dizininizin adı da Kubra değilse muhtemelen farklı bir yola sahip olurdunuz. u anda oturum açmış olan kişinin kullanıcı dizinini temsil etmek için `~` kısayolunu da kullanabilirsiniz: `~/Desktop/report.Rmd`.

6.2.2 Çalışma Dizini

- Tüm dosyalarınız nereye koymalısınız? Genellikle tek bir proje için tüm komut dosyalarınız ve veri dosyalarınız bilgisayarosundaki tek bir klasörde, o projenin çalışma dizininde olmasını istersiniz. Dosyalar bu ana proje dizini içindeki alt dizinlerde düzenleyebilirsiniz, örneğin tüm ham veri dosyaların **data/import** adlı bir dizine koyabilir ve tüm görüntü dosyaların **images** adlı bir dizine kaydedebilirsiniz.
- Kodunuz, uygun biçimi kullanarak yalnızca üç tür konumdaki dosyalar kullanılmaktadır.

Yer	Örnek
web	<code>"https://atalay-k.github.io/OLC731/import/veri1.txt"</code>
dizin içinde	<code>"veri1.txt"</code>
alt dizin içinde	<code>"import/veri1.txt"</code>

Bir komut dosyasında asla çalışma dizininizi ayarlamayın veya değiştirmeyin.

- R Markdown dosyalar otomatik olarak **.Rmd** dosyasının bulunduğu dizini çalışma dizini olarak kullanacaktır.
- Kodunuz çalışma dizininizin bir alt dizinindeki bir dosyaya ihtiyaç duyuyorsa (örneğin, `import/veri1.txt`), çalışma dizinini başka bir konuma veya bilgisayara taşındığında erişilebilir olması için dosyaya göreli bir yol kullanarak yükleyin:

```
dat <- read.table("import/veri1.txt") # doğru
```

Bu dosyaya aadaki gibi mutlak yol/adres ile yüklemeyin:

```
dat <- read.table("c:/Users/Kubra/Desktop/OLC731/import/veri1.txt") # yanlış
```

- Örnek veriyi düzgün aktarmak aadaki kodla yapılır.

```
(veri1 <- read.table("import/veri1.txt",
  header= TRUE,
  sep= ";",
  dec= ",",
))
```

```
##      no m_1  m_2 m_3  m_4 m_5
## 1 522  12 14.0  16 20.0  10
## 2 222   5   NA  20 10.0  10
## 3 454   5 10.2   6  4.0  10
## 4 567  10 20.0  NA 12.2  20
```

Ayrıca, Windows'a özgü geriye doru eik çizgi kullanma kuralının aksine, ileriye doru eik çizgi kullanma kuralına dikkat edin. Bu, dosyalara yapılan referansların iletim sistemlerinden bamsz olarak herkes için çalışmasını sağlamak içindir.

6.2.3 Nesneleri Adlandırma

Dosyalar, hem insanların hem de bilgisayarların kolayca bulabileceği şekilde adlandırılır. Te bazı önemli ilkeler:

- dosya ve dizin adları yalnızca harf, rakam, tire ve alt çizgi içermeli, dosya adı ve uzantısı arasında nokta (.) olmalıdır (bu bölük olmadıkça anlamna gelir!)
- Büyük harf kullanımı konusunda tutarlı olun (hatırlamayla kolaylaştırmak için bir kural belirleyin, örneğin her zaman küçük harf kullanın)
- dosya adının bölümlerini ayırmak için alt çizgi (_) ve bir bölümdeki sözcükleri ayırmak için tire (-) kullanın
- dosyalar mantıklı bir sırayla alfabetik hale getirin ve aradığınız dosyayı bulmanızla kolaylaştıran bir kalple adlandırın
- bir dosya adının listenin en üstüne tamak için önüne alt çizgi ekleyin veya sıraların kontrol etmek için tüm dosyaların önüne say ekleyin
- tarihler için YYYY-MM-DD biçimini kullanın, böylece kronolojik sıraya göre sıralanabilir

Örneğin, bu dosya adları tam bir karmaşıklık:

- Veri (Katilimci) 04-15.xls
- final_raporu2.doc
- Katilimci Veri Nisan 12.xls
- proje_notlari
- Anket Veri Kasım 15.xls
- rapor.doc
- rapor_son.doc

Benzer dosyaların aynı yapıya sahip olması ve dosyalar taramanın veya ilgili dosyaları bulmak için kod kullanmamasının kolay olması için dosyalar yapılandırılabilir.

- proje-notlari
- veri_katilimci_2021-04-12.xls
- veri_katilimci_2021-04-15.xls
- veri_anket_2021-04-15.xls
- rapor_v1.doc
- rapor_v2.doc
- rapor_v3.doc

Yukarıdaki dosyaları adlandırmak için baka yollar düşünün. Kendi proje dosyalarınızdan bazılarına bakın ve neleri geliştirebileceğinizi görün.

6.2.4 Yeni bir projeye başlamak

- Artık dosya sisteminin nasıl çalıştığını ve komut dosyalarının bunlara erişmesini kolaylaştırmak için nesneleri nasıl adlandıracağımızı anlamıza göre, projemizi yapmaya hazırız.

- Öncelikle, bu snf için tüm materyallerinizi tutacanz yeni bir dizin oluturun (benimki Ranaliz adında). Bu dizini Global Options’ın general bölümü altında varsaylan çalma dizini olarak ayarlayabilirsiniz. Bu, bir projede çalmıyorsanız dosyaların varsaylan olarak buraya kaydedilecei anlamna gelir.

Bu dizin OneDrive’daysa veya tam dosya yolu özel karakterler içeriyorsa ya da baz Windows makinelerinde 260 karakterden fazlaysa bazen sorunlara neden olabilir.

- Ardından, OLC731_2023 adında yeni bir proje oluturmak için **File** menüsü altında **New Project...** öesini seçin. Yeni oluturduunuz dizinin içine kaydettiinizden emin olun. RStudio kendini yeniden balatacak ve çalma dizini olarak bu yeni proje dizini ile açlacaktr.

Proje dizininin içeriini görmek için sa alt bölmedeki Files sekmesine tklayn. Tüm proje bilgilerini içeren Ranaliz.Rproj adl bir dosya göreceksiniz, projeyi açmak için üzerine çift tklayabilirsiniz.

Ayarlarınza bal olarak, özel kullanc ayarlarınz içeren .Rproj.user adl bir dizin de görebilirsiniz. Bu ve nokta ile balayan dier “görünmez” dosyalar yok sayabilirsiniz.

6.3 R Markdown

- Bu derste, bir içindekiler tablosu, uygun balklar, kod parçalar, tablolar, resimler, satr içi R ve bir kaynakça içeren bir R Markdown belgesi oluturmay öreneceiz.

R Markdown’a çok benzeyen quarto adında yeni bir tür tekrarlanabilir rapor format var. Bu derste quarto kullanmayacaz çünkü ayn anda hem quarto hem de R Markdown öreniyorsanz kafa kartrc olabilecek birkaç küçük fark var, ancak R Markdown’ örendikten sonra quarto’yu çok kolay bir ekilde örenebilirsiniz.

- Tekrar üretilebilir raporlar oluturmak için metin ve kodun kartrlmasn salayan R Markdown kullanacaz. Yeniden üretilebilir bir komut dosyas, kod bloklarında kod bölümleri içerecektir. Bir kod blou arka arkaya üç backtick sembolü ile balar ve biter, küme parantezleri arasında kod hakkında baz bilgiler bulunur, örnein {r chunk-name, echo=FALSE} (bu kodu çaltrr, ancak derlenen belgede kod blounun metnini göstermez). Kod bloklarının dndaki metin, balklar, paragraflar, listeler, kalnlatırma ve balantlar gibi biçimlendirmeyi belirtmenin bir yolu olan markdown ile yazlr. Örnek Dosyay beraber inceleyelim.
- Bir ablondan yeni bir R Markdown dosyas açarsanz, içinde birkaç kod blou bulunan örnek bir belge görürsünüz. Bir R Markdown (Rmd) belgesinden HTML veya PDF raporu oluturmak için belgeyi derlersiniz. Bir belgeyi derlemeye RStudio’da örme denir. Dosyanz bir rapora derlemek için üzerine tkladnz, içinden ineler geçen bir iplik yumana benzeyen bir düme vardır.

File > New File > R Markdown... menüsünden yeni bir R Markdown dosyas oluturun. Bal ve yazar deitirin, dosyay Ornek1.Rmd olarak kaydedin, ardından bir html dosyas oluturmak için ör dümesine tklayn.

6.3.1 YAML Header

- YAML bal çeitli seçenekleri ayarlayabileceiniz bölümdür.

```
---
title: "Demo"
author: "Kubra"
output:
```

```
html_document:
  toc: true
  toc_float:
    collapsed: false
    smooth_scroll: false
    number_sections: false
---
```

Seeneklerin ne ie yaradn grmek iin deerleri **false** tan **true** ya deitirmeyi deneyin.

- Varsaylan temalar unlardr: default, cerulean, cosmo, darkly, flatly, journal, lumen, paper, readable, sandstone, simplex, spacelab, united ve yeti. Linkten daha fazla temay inceleyebilirsiniz..

6.3.2 Kurulum

- Varsaylan ablonu kullanarak RStudio’da yeni bir R Markdown dosyas oluturduunuzda, otomatik olarak bir kurulum blogu oluturulur.

```
knitr::opts_chunk$set(echo = TRUE)
```

- Kod paralar iin daha fazla varsaylan seenei buradan ayarlayabilirsiniz. Olas seeneklerin aklamalar iin knitr seenekleri belgelerine bakn. knitr dokuman

```
knitr::opts_chunk$set(
  fig.width = 8,
  fig.height = 5,
  fig.path = 'images/',
  echo = FALSE,
  warning = TRUE,
  message = FALSE,
  cache = FALSE
)
```

Yukardaki kod aadaki seenekleri ayarlar:

- fig.width = 8 : varsaylan ekil genilii 8 intir (bunu tek tek ekiller iin deitirebilirsiniz)
- fig.height = 5 : varsaylan ekil ykseklii 5 intir
- fig.path = ‘images/’ : ekiller “images” dizinine kaydedilir
- echo = FALSE : ilenen belgede kod paraların gsterme
- warning = FALSE : herhangi bir ilev uyarı gsterme
- message = FALSE : herhangi bir ilev mesaj gsterme
- cache = FALSE : her rg rdnzde tm grntleri ve nesneleri oluturmak iin tm kodu altrn (zaman al kodunuz varsa TRUE olarak ayarlayn) Konsola str(knitr::opts_chunk\$get()) yazarak geerli kod blou seeneklerinin bir listesini bulun.

htiyacnız olan paketleri library()kullanarak da bu blounuza ekleyebilirsiniz. Genellikle bir komut dosyas zerinde alrken, baka bir eklenti paketi yklemeniz gerektiini fark edersiniz. library(...) arsn kodun en altına gmmeyin. En ste koyun, bylece kullanc hangi paketlerin gerekli olduuna dair genel bir fikir elde edilir.

tidyverse paketinden fonksiyon kullanacaz, bu yzden kurulum blounuza ykleyin.

6.3.3 Yap

- Bir içindekiler tablosu (`toc`) eklerseniz, bu tablo belge balklarınızdan oluşturulur. Markdown'daki balklar, balk balnın önüne bir veya daha fazla hash (`#`) eklenerek oluşturulur.
- Kendi analiz komut dosyalarınız getirirken aadaki yapıyı kullanın:
- Kullanmanız gereken tüm eklenti paketlerini yükleyin
- herhangi bir özel fonksiyon tanımlayın
- birlikte çalışacak verileri yükleyin veya simüle edin
- kaydetmeniz gereken her eyi kaydedin

Varsayılan metni silin ve balklar ve alt balklar oluşturarak belgenize biraz yap ekleyin. Baz verileri yükleyeceğiz, bir özet tablo oluşturacağız, verileri çizeceiz ve analiz edeceğiz.

6.3.4 Kod Bloklar

- Metninize eklemek için görüntüler, tablolar veya hesaplamalar oluturan ve görüntüleyen kod parçalar ekleyebilirsiniz. Baz verileri yükleyerek balayalım.
- İlk olarak, belgenizde bir kod bloğu oluturun. Bu kod iris veri setini yükler.

```
library(datasets)
data(iris)
```

6.3.5 Yorumlar

- Kod bloklar içine hash sembolü (`#`) ile yorum ekleyebilirsiniz. R yorumlayıcı, hash'ten satır sonuna kadar olan karakterleri yoksayacaktır.

```
n <- nrow(iris) # toplam satır sayısı
mu <- mean(iris$Petal.Length) # taç yaprak uzunluğu ortalaması
sd <- sd(iris$Petal.Length) # taç yaprak uzunluğu standart sapması

simule_deger <- rnorm(n, mu, sd)
```

- Bir kod parçasını, özellikle kod rapor metninde açıklanmıyorsa, orada ne yaptığınızı açıklayan bir yorumla balatmak genellikle iyi bir uygulamadır.
- Nesnelerinizi açık bir şekilde adlandırmıyorsanız, genellikle açıklayıcı yorumlar eklemeniz gerekmez. Örneğin, yukardaki üç nesneyi `n_iris`, `ort_petal` ve `sd_petal` olarak adlandırmı olsaydı, yorumlar atlardı.
- Yorumların bir baka kullanımı da çalrtmak istemediğiniz ancak silmek de istemediğiniz bir kod bölümünü "yorumlamak". Örneğin, bir paketi yüklemek için kullanılan kodu kodunuza dahil edebilirsiniz, ancak kodun her çaltrıldığında uzun bir yüklemeye zorlamaması için her zaman yorumlamımanız gerekir.

```
# install.packages("dplyr")
# install.packages("tidyr")
# install.packages("ggplot2")
```

Satırlar seçip `Cmd-shift-C` (Mac) veya `Ctrl-shift-C` (Windows) yazarak aynı anda birden fazla satıra yorum yazabilir veya yorumu kaldırabilirsiniz.

- Kodunuzu iyi bir eklede yorumlamak biraz sanattır. Bu beceriyi gelitirmenin en iyi yolu, bakaların kodların okumak ve bakaların kodunuzu incelemesini salamaktır.

6.3.6 Satr ç kodlar

- imdi setosa ve virginica çiçek türlerinde yaprak uzunlukların analiz edelim. Önce analiz kodunu çalıştracaz. Daha sonra makalemizde kullanmak isteyebileceimiz sayılar deikenlere kaydedeceiz ve uygun eklede yuvarlayacaz. Son olarak, bir sonuçlar biçimlendirmek için glue::glue() kullanacaz.

```
# analiz
library(dplyr)
setosa_petal <- filter(iris, Species == "setosa") %>% pull(Petal.Length)
virginica_petal <- filter(iris, Species == "virginica") %>% pull(Petal.Length)
petal_test <- t.test(setosa_petal, virginica_petal)

# rapor edilecek degerleri yorumlama
t <- petal_test$statistic %>% round(2)
df <- petal_test$parameter %>% round(1)
p <- petal_test$p.value %>% round(3)
# handle p-values < .001
p_symbol <- ifelse(p < .001, "<", "=")
if (p < .001) p <- .001

# sonuclar birletirme
petal_result <- glue::glue("t = {t}, df = {df}, p {p_symbol} {p}")
```

- Sonuçlar, aadaki gibi görünen satr içi R kodu ile bir paragrafa ekleyebilirsiniz:

virginica çiçeklerinin yapraklar setosa çiçeklerinin yapraklarından uzundur (t = -49.99, df = 58.6, p < 0.001).

6.3.7 Tablolar

- Çalmaların yöntem bölümüne betimsel bilgiler eklemek istediimizde

```
ozet_tablo <- iris %>%
  group_by(Species) %>%
  summarise(
    n = n(),
    ortalama = mean(Petal.Length),
    sd = sd(Petal.Length)
  )
ozet_tablo
```

```
## # A tibble: 3 x 4
##   Species      n ortalama    sd
##   <fct>    <int>   <dbl> <dbl>
## 1 setosa     50    1.46 0.174
## 2 versicolor 50    4.26 0.470
## 3 virginica  50    5.55 0.552
```

Yukardaki tablo etkileimli görünümde tibble biçiminde yazdırılacak, ancak ördüğünüzde YAML balndaki df_print ayarındaki biçimi kullanacaktır.

- Yukardaki tabloda, sütun etiketlerini deitirerek, ortalamalar yuvarlayarak ve bir balk ekleyerek daha okuyucu dostu hale getirilebilir. Bunun için knitr::kable() ilevini veya tabloların biçimlendirmek için dier paketlerdeki daha özel ilevleri kullanabilirsiniz.