1. Aby załadować pakiet do R, należy wywołać funkcję library z argumentem będącym nazwą pakietu, np library(grid)
2. Aby zainstalować wybrany pakiet, należy wpisać w konsoli R następujące polecenie install.packages("nazwa pakietu")
3. Dołączenie bazy do R attach(Orange), odłączenie detach(Orange)
4. Funkcja c() umożliwia tworzenie wektora z pojedynczych elementów oraz innych wektorów c(1,5,7,9,12), c(1, 4, 6, "zielony", "niebieski", "biały")
5. Każdemu obiektowi w R można przypisać nazwę. Do pobierania nadawania nazwy elementom danego obiektu służy funkcja names() np: names(kolory) = c("zielony", "niebieski")
6. Sekwencje(ciągi), będące elementami wektora można tworzyć za pomocą operatora : bądź funkcji seq(). 30:10, seq(1,45,by=3), c(seq(from=1, by=3, length=15),1:10)
7. Środowisko R umożliwia replikację, a więc powielanie danych.Służy do tego funkcja rep(). Jej pierwszym argumentem jest obiekt (najczęściej wektor), który ma zostać powielony. Kolejne(opcjonalne) argumenty to następujące nieujemne liczby całkowite:
   1. times - ilość powtórzeń obiektu,
   2. length.out - długość wektora wynikowego,
   3. each - ilość powtórzeń każdego elementu wektora źródłowego.
   4. Przykład:
      1. rep(5:8,4),
      2. rep(5:8,each=4) # inny wynik niż poprzednio!,
      3. rep(5:8,length.out=10)
8. Lista to typ danych, który pozwala łączyć obiekty różny chtypów. Konstruktorem listy jest funkcja list().
   1. przedmiot <-c ( "piłka" , "skrzynia" )
   2. kolor <- c( "czerwony","zielony" )
   3. waga <- c( 2,8 )
   4. lista <- list(przedmiot, kolor, waga)
   5. lista$waga[which(lista$przedmiot=="piłka")]
9. Wektor można posortować w kolejności rosnącej bądź malejącej.Służy do tego funkcja sort()
   1. sort(kolory) # domyślnie sortujemy w kolejności rosnącej
   2. sort(kolory, decreasing = TRUE) # zmieniamy kolejność na malejącą
10. Import danych z plików zewnętrznych możliwy jest w R dzięki funkcji read.table(), która czyta plik w formacie tabeli i tworzy na jego podstawie ramkę danych. Każda obserwacja odpowiada linii w pliku, a zmienna - polu,czyli kolumnie.Ważniejsze parametry:
    1. file - pierwszy parametr, określa nazwę pliku. Jeżeli nie jest podana pełna ścieżka, plik jest pobierany z katalogu roboczego R,
    2. header: TRUE/ FALSE - określa, czy nagłówek (nazwy kolumn) znajdują się w pierwszym wierszu, domyślnie ustawiony na FALSE,
    3. sep- definiuje separator pól, czyli znak pomiędzy kolejnymi wartościami w obrębie obserwacji, np. ”;”, ”,” (domyślnie brak: ””),
    4. dec - definiuje znak używany do oddzielania części dziesiętnych,domyślnie ”.”.
    5. Przykład: read.table("warzywniak.txt", header=TRUE)
11. Analogiczna funkcja: read.csv służy do czytania plików z rozszerzeniem .csv.
12. Dane analizowane podczas pracy można zapisać w pliku za pomocą funkcji write.table(). Dane niebędące ramką danych ani macierzą są przed zapisem konwertowane do formatu ramki. Ważniejsze parametry:
    1. x - pierwszy parametr, określa obiekt, który chcemy zapisać,file- nazwa pliku, w którym mają być zapisane dane. Jeżeli nie jest podana pełna ścieżka, plik jest zapisywany do katalogu roboczego, ””oznacza wydruk w konsoli R
    2. append: TRUE/FALSE - umożliwia dopisanie danych do istniejącego już pliku, jeżeli ustawiony na TRUE. W przeciwnym przypadku plik o tej samej nazwie zostanie zniszczony. Domyślnie ustawiony na FALSE,
    3. sep - definiuje separator pól, czyli znak pomiędzy kolejnymi wartościami w obrębie obserwacji, np. ”;”, ”,” (domyślnie spacja: ” ”),
    4. dec - definiuje znak używany do oddzielania części dziesiętnych,domyślnie ”.”,
    5. Przykład: write.table(dane, file = "dane.txt", sep=",")
13. Środowisko R umożliwia definiowania własnych funkcji. Odbywa się to poprzez przypisanie nazwy funkcji do jej definicji:
    1. nazwa <- function(arg1, arg2, ...) wyrażenie
    2. nazwa <- function(arg1, arg2, ...){blok wyrażeń}
14. Środowisko R umożliwia konstrukcję pętli, czyli cyklicznego wykonywania ciągu instrukcji określoną liczbę razy, dla każdego elementu zbioru. Odbywa się to w bloku:
    1. for(zmienna in zbiór) instrukcja
    2. for(zmienna in zbiór){ciąg instrukcji}
15. Środowisko R umożliwia konstrukcję instrukcji warunkowych, które pozwolą na wykonanie różnych obliczeń w zależności od tego, czy zdefiniowane wyrażenie logiczne jest prawdziwe, czy też nie. Odbywa się to w bloku:
    1. if(warunek) instrukcja
    2. if(warunek) instrukcja else instrukcja
16. Zastosowanie pętli w poprzednim przykładzie nie jest najbardziej efektywnym rozwiązaniem. Istnieje jednak w R instrukcja warunkowa umożliwiająca pracę na całych wektorach: ifelse(warunek, instrukcja gdy TRUE, instrukcja gdy FALSE)
    1. Przykład:
       1. ifelse(circumference>średnia, "Obwód większy niż średnia",ifelse(circumference==średnia,"Obwód równy średniej", "Obwód Mniejszy niż średnia"))
17. W R mamy również możliwość konstrukcji pętli innego rodzaju:while repeat. Pętla while Wykonuje polecenia dopóty, dopóki warunek jest spełniony. Pętla repeat Wykonuje instrukcje do momentu, gdy ją wprost zatrzymamy(break).
    1. Przykład: while (i<=10){ inst }
18. Analizę danych często ułatwia ich skategoryzowanie poprzez stworzenie tzw.szeregu rozdzielczego. W R służą do tego funkcje table() oraz xtabs() Przykład:
    1. t1=table(dane)
    2. t2=xtabs(dane)
19. Liczbę przedziałów można wybrać korzystając np.ze wzoru k ~'√n, gdzie oznacza liczbę obserwacji, lub samemu określając granice przedziałów. Funkcją przydatną w tej sytuacji jest cut(). Przykład
    1. y1=cut(x, sqrt(n))
    2. table(y1)
    3. y2=cut(x,breaks=c(205,215,225))
20. Zmienne znajdujące się w bazie znajdujące możemy uzyskać po wywołaniu polecenia: names(nazwa\_bazy)
21. Funkcja length() pozwala wyznaczyć wielkość analizowanej próby
22. Funkcje, które pozwalają wyznaczać najpopularniejsze statystyki opisowe są następujące:
    1. mean(petal) - średnia arytmetyczna; dodatkowy parametr trim∈(0,0.5) pozwala na obcięcie po trim\*100% skrajnych obserwacji z każdego końca próbki, przed obliczeniem średniej(przydatne, gdy występują obserwacje odstające),
    2. var(petal) - wariancja próbkowa
    3. sd(petal) - odchylenie standardowe (pierwiastek z wariancji próbkowej)
    4. min(petal)- minimum z próby
    5. max(petal)- maksimum z próby
    6. range(petal)- przedział zmienności próby:(min,max)
23. Funkcje, które pozwalają liczyć statystyki opisowe związane z kwantylami empirycznymi to:
    1. median() - mediana próbkowa
    2. quantile() - kwantyle próbkowe, ich rzędy podawane są w wektorze będącym drugim argumentem funkcji np: quantile(petal,c(0.1,0.6,0.9))
    3. IQR() - rozstęp międzykwartylowy.
24. Aby obliczyć dominantę trzeba wykonać następujące polecenia:
    1. skategoryzować dane w szereg rozdzielczy tablica<-table(dane)
    2. uporządkować szereg rozdzielczy względem liczebności kategorii, w kolejności od najbardziej licznej do najmniej licznej: tablica\_s<-names(sort(tablica,decreasing=T))
    3. zwrócić nazwę kategorii, która znajduje się na pierwszym miejscu: dominanta = tablica\_s[1]
25. Funkcja summary() umożliwia podsumowanie charakterystyka obiektu, który jest argumentem funkcji. Może ona dać inny wynik dla zmiennej numerycznej, inny dla jakościowej, a inny np. dla ramki danych.
26. Dla obliczenia momentów empirycznych służy pakiet moments. Zawiera on też inne przydatne statystyki, m.in.
    1. moment(x, order = 1, central = FALSE, absolute = FALSE, na.rm= FALSE)- umożliwia obliczenie momentów zwykłych (domyślnie),centralnych (jeżeli central=TRUE), absolutnych (jeżeli absolute=TRUE),oraz centralnych absolutnych (jeżeli oba parametry jednocześnie:absolute oraz central ustawione są na TRUE), ustalonego rzędu(parametr order)
    2. skewness() - umożliwia obliczenie współczynnika skośności
    3. kurtosis() - umożliwia policzenie kurtozy rozkładu.
    4. Kurtoza liczona wg powyższego wzoru dla danych z rozkładu normalnego wynosi ok. 3.
27. Do wylosowania próbki na podstawie zadanego wektora danych służy funkcja sample(). Argumenty są następujące:
    1. pierwszym jest albo wektor źródłowy, z którego będziemy wybierać, albo liczba całkowita, określająca rozmiar próby źródłowej,
    2. rozmiar próby, którą chcemy uzyskać,
    3. replace - określa, czy pobieramy z powtórzeniami (=TRUE),czy bez (=FALSE, domyślnie),
    4. prob - opcjonalny wektor prawdopodobieństw wyboru poszczególnych elementów z wektora źródłowego
    5. Przykłady zastosowania:
       1. sample(LETTERS[1:26], 10, replace=TRUE)
       2. sample(letters[1:26], 5) bez powtórzeń
       3. sample(1000, 20, replace=TRUE)
28. W R dostępny jest pakiet funkcji statystycznych (stats), które umożliwiają pracę ze zmiennymi losowymi pochodzącymi z różnych rozkładów.
    1. Najczęściej przydatne są funkcje, których nazwa składa się z przedrostka:
       1. r(random generation) - pozwala wygenerować próbkę losową pochodzącą z danego rozkładu
       2. d(density) - pozwala wyznaczyć wartość gęstości w punkcie dla danego rozkładu,
       3. p(probability distribution) - pozwala wyznaczyć wartość dystrybuanty punkcie dla danego rozkładu,
       4. q(quantile) - pozwala wyznaczyć wartość funkcji kwantylowej w punkcie dla danego rozkładu,
    2. oraz nazwy rozkładu prawdopodobieństwa, np.:
       1. norm - rozkład normalny,
       2. unif - rozkład jednostajny,
       3. exp - rozkład wykładniczy,
       4. gamma - rozkład gamma,
       5. beta - rozkład beta,
       6. chisq - rozkład Chi-kwadrat,
       7. t - rozkład t-Studenta
29. W R można generować wykresy m.in. dystrybuanty i gęstości znanych rozkładów. Służy do tego np. funkcja plot() np: plot(dnorm,-4,4, col="red"), inna funkcja do wykresów to np: curve(pexp(x), add=TRUE, col="red")
30. Przydatne funkcje powiązane z dystrybuantą empiryczną
    1. ecdf(x) - pozwala wyznaczyć dystrybuantę empiryczną dla wektora obserwacji x,
    2. plot.ecdf() - pozwala narysować dystrybuantę empiryczną
31. Kolejną przydatną funkcją jest knots(), która pozwala wyznaczyć wektor punktów skoku dystrybuanty empirycznej.
32. Funkcja qqnorm() pozwala wygenerować wykres porównujący kwantyle empiryczne próby z kwantylami rozkładu normalnego. Ponadto, aby narysować na nim linię prostą, która przechodzi przez górny i dolny kwartyl, należy wywołać funkcję qqline().
33. Funkcja qqplot() pozwala wygenerować wykres umożliwiający porównanie kwantyli empirycznych dwóch prób.
34. Wiele poleceń generujących wykresy posiada wspólne argumenty,m.in.
    1. height, width - wysokość i szerokość wykresu,
    2. col - określa kolor elementów wykresu (tj. słupków, wycinków koła itp.),
    3. main, sub - tytuł i podtytuł wykresu,
    4. axes, axisnames - odpowiada za wyświetlanie i opisanie osi wykresu, domyślnie: TRUE,
    5. xlab, ylab - umożliwia wprowadzenie własnego tekstowego opisu osi x i y wykresu,
    6. add - umożliwia wyświetlenie wykresu wspólnie z innym,wcześniej wygenerowanym wykresem, domyślnie: FALSE.
35. W środowisku R dostępna jest bogata gama kolorystyczna. Lista 657 predefiniowanych kolorów jest dostępna po wywołaniu funkcji
    1. colors()
    2. col=rainbow(5)- elementy wykresu będą kolorowane naprzemiennie pięcioma kolorami tęczy,
    3. col=45- wszystkie elementy wykresu będą w jednakowym odcieniu koloru niebieskiego (”cadetblue 3”),
    4. col=c(93,2,"darkgrey")- elementy wykresu będą kolorowane naprzemiennie trzema kolorami
36. Dodanie legendy do wykresu jest możliwe poprzez wywołanie funkcji legend(). Położenie legendy na wykresie może być zrealizowane poprzez:
    1. argument x(lub x,y) zawierający współrzędne położenia lub tekst:”bottomright”, ”bottom”, ”bottomleft”, ”left”, ”topleft”, ”top”,”topright”, ”right”, ”center”.argument locator(1)- umożliwia umieszczenie legendy na wykresie we wskazanym myszką miejscu.
    2. Pozostałe ważne parametry:
       1. bty - określa czy legenda ma być ̇c obramowana; możliwe wartości: ”o” -ramka (domyślnie), lub ”n” - brak ramki,
       2. bg - jeżeli bty=”o”, to pozwala ustawić tło ramki z legendą,
       3. cex - określa wielkość czcionki w legendzie,
       4. fill - pozwala wyświetlić kwadraciki przy opisie i przypisać im wybrane kolory, np.fill=rainbow(4).
    3. Funkcję legend() należy wywołać po tym, jak R wyświetli wykres.
37. Podstawową i najbardziej uniwersalną funkcją generującą wykresy R jest funkcja plot(). Może ona dawać różne rezultaty, w zależności od podanych parametrów. Najważniejszym parametrem funkcji plot jest type - typ wykresu, jeden z:
    1. "p"- punktowy,
    2. "l"- liniowy,
    3. "b"- punktowo-liniowy,
    4. "s"- schodkowy,
    5. "S"- schodkowy, odwrócony,
    6. "h"- kreskowy (”histogramowy”).
38. Funkcja par() wywołana przed funkcją plot() umożliwia przekazanie bądź pobranie parametrów graficznych. Często Używane argumenty:
    1. mfcol,mfrow- wektor określający liczbę wierszy i kolumn,które określają tablicę, na której rysowane będą poszczególne wykresy
39. Funkcja barplot() pozwala wygenerować pionowy (domyślnie) lub poziomy wykres słupkowy.
    1. W zależności od postaci argumentu głównego, możliwe są następujące rodzaje wykresów słupkowych:
       1. obrazujący wartości poszczególnych obserwacji - jeden słupek odpowiada jednej obserwacji; jako argument główny podajemy nazwę zmiennej, dla której chcemy zilustrować wartości obserwacji,
       2. obrazujący wartości poszczególnych obserwacji - jeden słupek odpowiada jednej obserwacji; jako argument główny podajemy nazwę zmiennej, dla której chcemy zilustrować wartości obserwacji,
    2. Parametry specyficzne dla funkcji barplot:
       1. horiz- poziome położenie słupków, domyślnie: FALSE
       2. beside- określa sposób ułożenia słupków w przypadku podziału ze względu na kategorie innej zmiennej. Jeżeli beside=TRUE, to słupki są zgrupowane obok siebie, jeżeli beside=FALSE(domyślnie), to słupki zestawione są jeden na drugim.
40. Funkcja hist() pozwala wygenerować histogram. Ważne parametry:
    1. breaks- parametr określający sposób wyznaczania klas na histogramie, może to być jeden z następujących argumentów:
       1. wektor zawierający punkty podziału,
       2. liczba określająca ilość klas,
       3. nazwa algorytmu, który wylicza liczbę klas,
       4. funkcja obliczająca liczbę klas.Parametr ten domyślnie ustawiony jest na ”Sturges” (nazwa algorytmu),
    2. freq- na osi pionowej odkładane są liczebności poszczególnych klas (domyślnie TRUE),
    3. prob- przeciwieństwo freq: na osi pionowej odkładane są częstości względne(prawdopodobieństwa empiryczne) dla poszczególnych klas (domyślnie FALSE)
41. Funkcja stem() pozwala wygenerować wykres łodyga-liście. Jest to wykres tekstowy ilustrujący wartości zmiennej. Ważne parametry:
    1. scale- określa skalę wykresu (domyślnie 1).
    2. width- szerokość wykresu (domyślnie 80),
42. Funkcja pie() pozwala wygenerować wykres kołowy.
43. Funkcja boxplot() pozwala wygenerować wykres skrzynkowy. Ważne parametry funkcji:
    1. range- określa zakres długości wąsów. Dodatnia wartość range(domyślnie wynosi ona 1,5) pomnożona przez IQR wyznacza granicę długości wąsów. Wartość 0 powoduje wygenerowanie wykresu skrzynkowego bez uwzględnienia obserwacji odstających.
    2. horizontal- wartość logiczna określająca, czy wykres ma być ułożony poziomo (domyślnie FALSE).
44. Do agregacji danych służy funkcja aggregate()
    1. aggregate(mtcars$mpg, by=list(mtcars$cyl), FUN=mean)
45. Testy t-Studenta to parametryczne testy dla jednej lub dwóch prób, polegające na testowaniu hipotezy dotyczącej wartości oczekiwanej (nazywane też testami istotności).Założenia stosowania:pomiary podlegają rozkładowi normalnemu bądź rozmiar próbki jest duży (z reguły wystarczy>30). Funkcja t.test() pozwala przeprowadzić test t-Studenta. Pierwszym argumentem funkcji jest wektor z danymi. Kolejne ważne(opcjonalne) argumenty są następujące:
    1. y - drugi wektor z danymi
    2. alternative : two.sided / less / greater - określa hipotezę alternatywną(domyślnie: dwustronna),
    3. mu - liczba określająca hipotetyczną wartość oczekiwaną rozkładu (bądź różnicę wartości oczekiwanych w przypadku dwóch prób), domyślnie równa 0
    4. paired - wektor logiczny określający, czy test ma być przeprowadzony dla danych powiązanych (zależnych), domyślnie FALSE. Jeżeli paired=TRUE,wówczas wektory z danymi muszą być równej długości
46. Standardowy test t-Studenta zakłada, że wariancje rozkładów obu próbek są zbliżone. Sprawdzimy więc najpierw, czy to ma miejsce.Stosujemy test na równość wariancji var.test(x1,x2)
47. Funkcja chisq.test() pozwala przeprowadzić test oparty na statystyce chi-kwadrat.Aby wykonać test chi-kwadrat zgodności, najpierw należy odpowiednio przygotować dane. Wartości obserwacji z próbki muszą być pogrupowane w klasy (szereg rozdzielczy punktowy bądź przedziałowy), aby obserwowane liczebności klas mogły zostać porównane z oczekiwanymi liczebnościami (wyznaczonymi przy założeniu, że dane pochodzą z testowanego rozkładu prawdopodobieństwa).
48. Funkcja ks.test() pozwala przeprowadzić jedno- lub dwu próbkowy test Kołmogorowa-Smirnowa zgodności rozkładów.
49. Test Shapiro-Wilka jest testem zgodności wyłącznie z rozkładem normalnym. Funkcja shapiro.test() pozwala przeprowadzić test Shapiro-Wilka
50. Funkcja prop.test() pozwala przeprowadzić test proporcji, jak też uzyskać przedział ufności dla tej proporcji. Ten test jest przybliżonym i opiera się na statystyce chi-kwadrat.Ważną opcją jest używanie (bądź nie)correct=T. Za pomocą correct=T(domyślnie) uzyskujemy tzw.poprawkę na ciągłość.Taka poprawka się stosuje, gdy przynajmniej jedna z liczebności oczekiwanych jest bardzo mała (można stosować już przy liczebnościach mniejszych niż 10,a na pewno przy mniejszych niż 5).