```r
remotes::install_github("gilberto-sassi/statBasics")
library(statBasics)
library(tidyverse)
```

```
Downloading GitHub repo gilberto-sassi/statBasics@HEAD

rlang     (0.4.11 -> 0.4.12) [CRAN]
lifecycle (1.0.0  -> 1.0.1 ) [CRAN]
stringi   (1.7.4  -> 1.7.5 ) [CRAN]
pillar    (1.6.2  -> 1.6.4 ) [CRAN]
tibble    (3.1.4  -> 3.1.5 ) [CRAN]
Installing 5 packages: rlang, lifecycle, stringi, pillar, tibble

Installing packages into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

✓  checking for file '/tmp/RtmpdUqNCK/remotes46d809a4/gilberto-sassi-statBasics-56f6
─  preparing 'statBasics':
✓  checking DESCRIPTION meta-information
─  checking for LF line-endings in source and make files and shell scripts
─  checking for empty or unneeded directories
   Omitted 'LazyData' from DESCRIPTION
─  building 'statBasics_0.1.0.tar.gz'

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

```r
n <- 100
num_amostra <- 1000
media_pop <- 10
calculos <- seq_len(num_amostra) |>
  map_dbl(\(k) {
    amostra <- rnorm(n, mean = media_pop, sd = 2)
    (mean(amostra)  - media_pop) * sqrt(n) / sd(amostra)
  })
calculos
```

0.105183069961567 · 1.44319867828186 · 0.158473074999574 · -1.08663232496378 · 1.531993956294
-0.168300829820724 · 0.754985621516972 · -1.22345894799884 · 0.0900857127419628 · -0.75516658
0.282041892889608 · -1.21339973820054 · 0.393777397326674 · 1.07506504113115 · -1.39222378732
-1.29328719636468 · -0.563985013406954 · 0.0215911748624674 · -1.2867875866899 · -0.3357746682
0.154022052262924 · 0.403898215234981 · 0.285342526705025 · 0.0795733916726533 · -0.435957114
-1.37547207835353 · 0.0453398727867129 · -1.57036390498853 · 0.502797522502557 · 0.5202180090
0.420237268095538 · -0.0325286419839884 · -0.270980650341935 · -0.621985545312523 · -0.8070541
-1.43843551211335 · 1.66109530187994 · 0.531579661920989 · -0.212532801342423 · 0.36397114667
-0.830410722907369 · -0.664153168478169 · -0.443777321614144 · -0.0398725564180929 · 0.0644456
1.19272090858559 · 1.05934310282579 · 0.788615844414051 · 0.598304356068148 · -0.08437653933
0.24075894209254 · -0.918278415896713 · 0.139186993145239 · 1.33047987075985 · -1.21434677374
2.23211300872834 · 1.78910117844047 · 0.0588750656734874 · -0.146099925815856 · -0.0486424241
-0.884082114183462 · 1.73256834233541 · -0.972211329018824 · -1.9156765391269 · 0.44189472388
0.485325987673163 · 1.39629825026195 · -1.78447322523163 · 0.178132600642166 · 0.07006697331
1.00480027381889 · 0.827998842730948 · -0.585009668669752 · 0.533907305559033 · -1.5470017169
0.788869314711295 · -0.867864899927686 · -0.369362255236357 · -0.0997258334430702 · -0.4995676
0.267717355775909 · 0.258782461299421 · 1.13566049740103 · 1.35755590276578 · 0.993772357402
-1.68426333286915 · 0.155844319374057 · -0.296515530839971 · -0.326932721072744 · 0.375204149
0.086833843723932 · 1.15339311916435 · 0.0526063185648161 · -1.15570986661957 · -0.2010589810
0.497116140945608 · -1.38666710651799 · 1.07142568651727 · 1.57706626978032 · -1.538189306697
-0.302307952420639 · 0.944083768625352 · 0.0383926484852538 · -0.78874261614286 · 0.229870122
-3.0239598626441 · 0.47099266199696 · 0.776164961261645 · -1.57262567158646 · -1.339512596506
-0.990418266398937 · 0.903750017316983 · 0.248695565859023 · 1.0263339189784 · -0.57823764235
-1.35789552384965 · 1.29190332059166 · 0.563136584848419 · 0.603976496744238 · -0.63622413182
-0.718725826915898 · 0.383824110132427 · 0.580799301659971 · 0.671291172104242 · 2.2747575129
0.723707548902977 · -0.898300001989527 · -0.628107478135195 · -0.389569353420619 · -2.74657806
0.215612912182186 · -1.78525443757722 · 1.29380462297303 · 0.192553610596206 · 0.51124803289
-1.14964957794406 · 0.907156138912171 · -0.444417318146627 · -0.674631197452035 · 1.017060768
0.811462619863409 · 0.272274924110142 · 0.781826715272073 · -0.227162804297108 · -0.653880635
-0.0872326850673182 · -0.914186204085035 · 0.722214841755185 · -1.20922885045086 · 0.84573004
0.578426892817073 · 0.222112406521308 · -0.240014010653008 · -0.420504141472873 · -1.24365908
-0.201134210117032 · 1.28462654201011 · 1.86486195127349 · -1.3570078675997 · 0.4588591290744
-1.02010601260587 · -0.136296908912202 · 0.364995984782065 · 0.767521130149961 · 0.6629682926
0.544843296813701 · -2.76915727385347 · 0.792304269424686 · 0.517476044874061 · 1.8071733538
-1.33656456244644 · 0.809439422256419 · 0.58907465895609 · -0.309399250615529 · 0.58877129369
-1.27114922844913 · 1.11462594930374 · -0.553115526674003 · 0.333926869773026 · -1.0322776402
-1.47734942872241 · 1.06178683462874 · 1.42521246511781 · 0.251186267839835 · 0.4119987458585
-0.553120141874023 · 1.10520397128806 · 0.581555801135438 · -1.57231879956745 · -1.6845527763
-0.26083968048862 · -1.6768060288034 · 0.303304770513005 · 0.663384365998075 · 0.354017850536
-0.87673923327999 · 0.387638380076895 · 1.47514267927805 · 0.213170447565809 · -0.23293517649

interepretação do intervalo de confinaça

Intervalo de confinaça t

população: N(10, 2)

- Construir intervalo de confiança quando **NÃO CONHECEMOS O DESVIO PADRÃO DA POPULAÇÃO.**
- coeficiente de confiança: 97%

```
num_amostras <- 1000
n <- 13
media_pop <- 10
dp_pop <- 2
```

```r
intervalos_corretos <- seq_len(num_amostras) |>
  map_dbl(\(k) {
    amostra <- rnorm(n, mean = media_pop, sd = dp_pop)
    quantil <- qt(0.985, df = n - 1)
    li <- mean(amostra) - quantil * sd(amostra) / sqrt(n)
    ls <- mean(amostra) + quantil * sd(amostra) / sqrt(n)
    return(li <= media_pop & media_pop <= ls)
  })
```

```r
intervalos_corretos
mean(intervalos_corretos)
```

```
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 ·
1 · 1 · 1 · 1 · 1 · … · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
```

```r
amostra <- rnorm(20, mean = 10, sd = 4)
```

```r
ci_norm(amostra, conf_level = 0.96)
```

A tibble: 1 × 3

lower_ci	upper_ci	conf_level
<dbl>	<dbl>	<dbl>
6.807489	10.58234	0.96