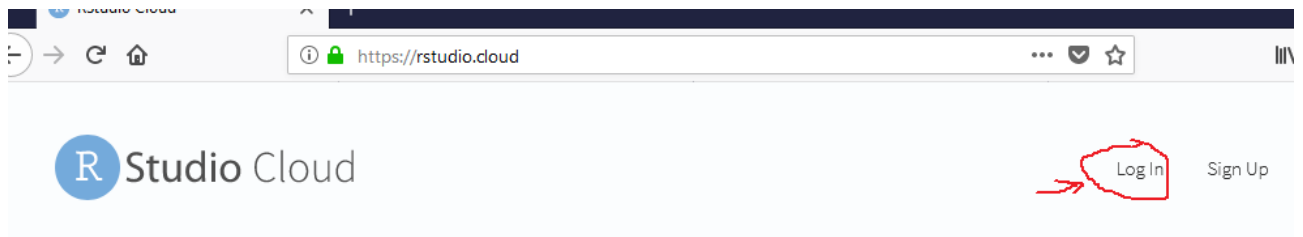


## Доданок. Робота з сайтом rstudio.cloud. Побудова гістограми

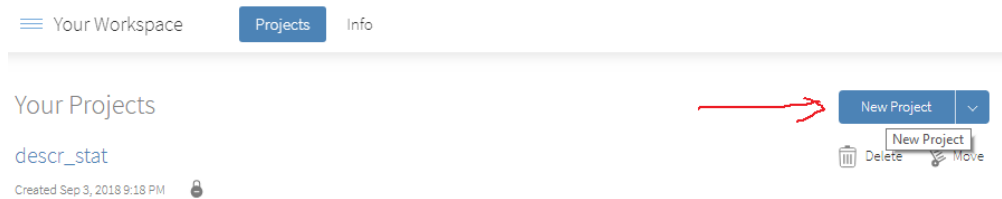
Якщо ви працюєте з сайтом rstudio.cloud, а не локальною копією програми rstudio, зареєструйтесь:



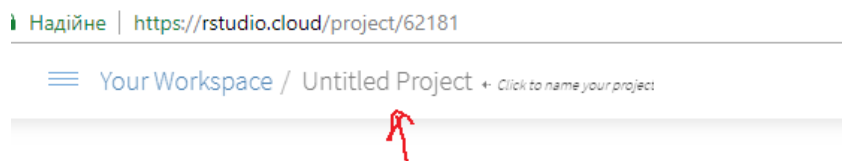
Або введіть логін



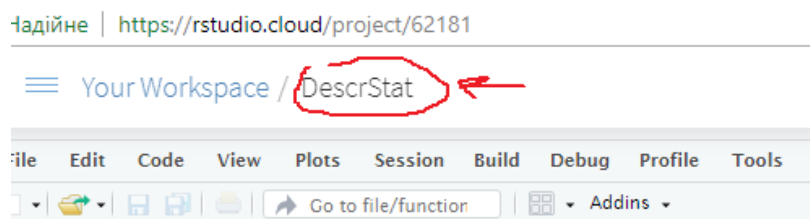
Якщо заходите перший раз, створіть новий проект:



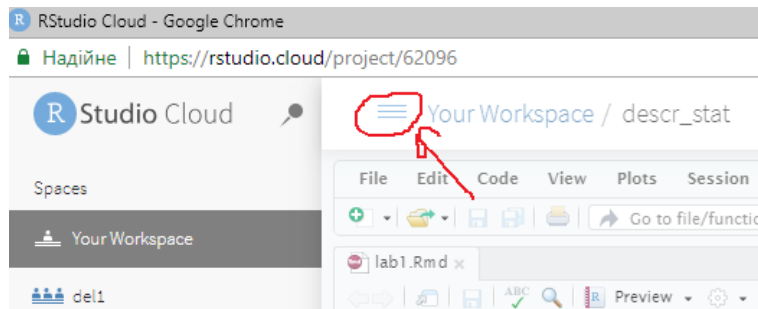
Назвіть його наприклад DescrStat:



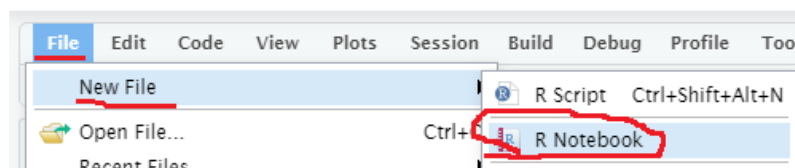
впишіть назву



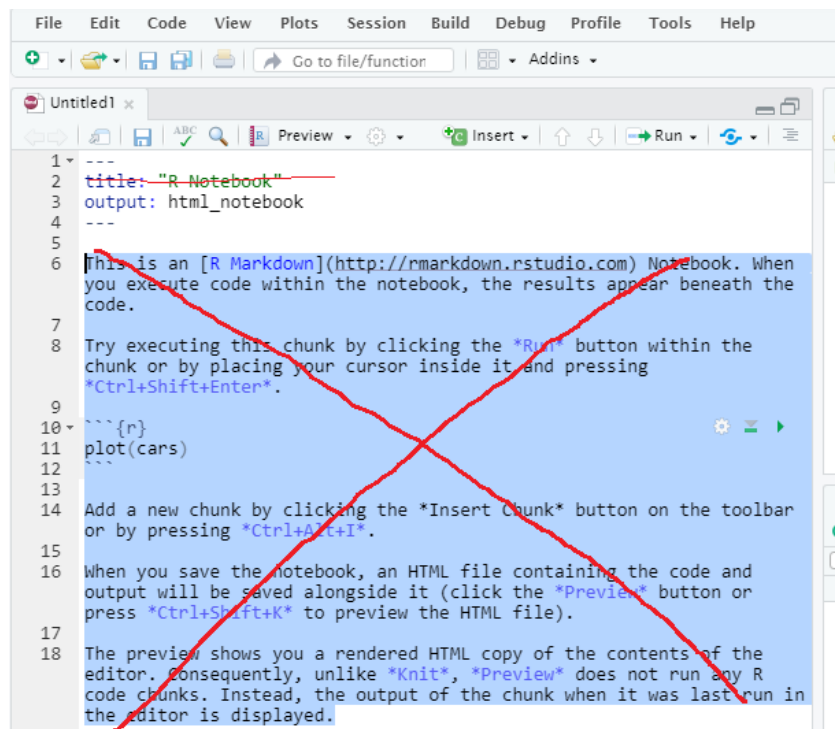
Сховайте браузер проектів



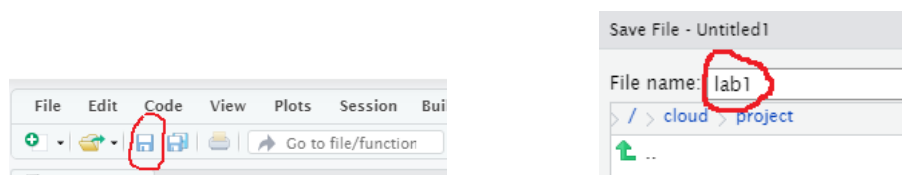
Створіть новий записник (він буде збережений в вашому новому проекті DescrStat)



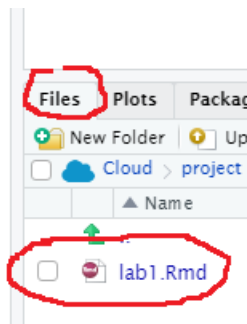
Видаліть у вікні редактора записника закреслений текст



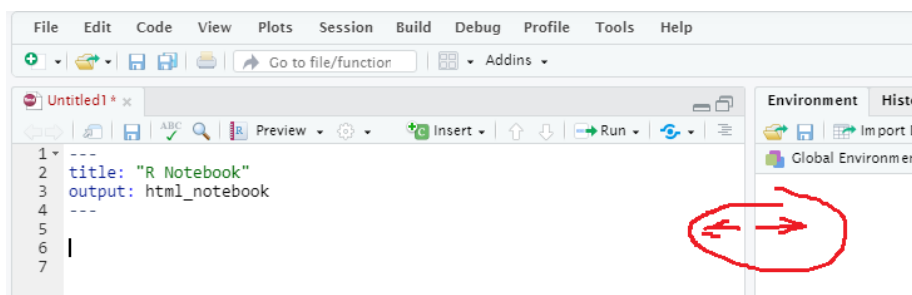
Збережіть ваш записник під ім'ям lab1 (комбінація клавіш Ctrl-S).



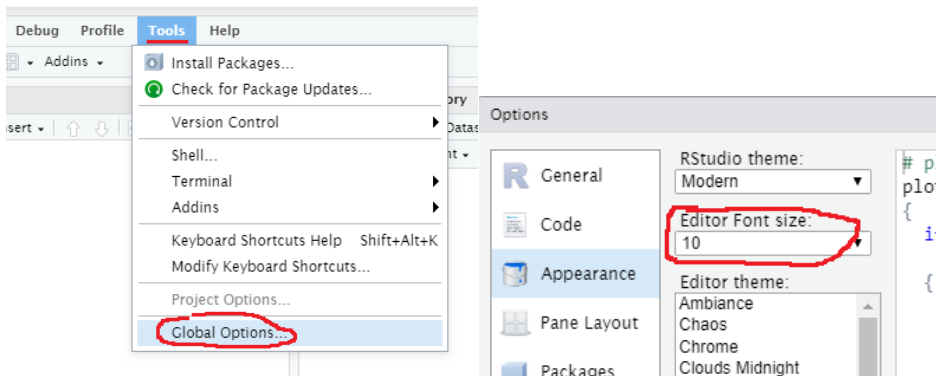
Зверніть увагу на вікно Files (файли поточного проекту), в ньому має з'явитися ваш файл. В цьому вікні можна відкривати файли.



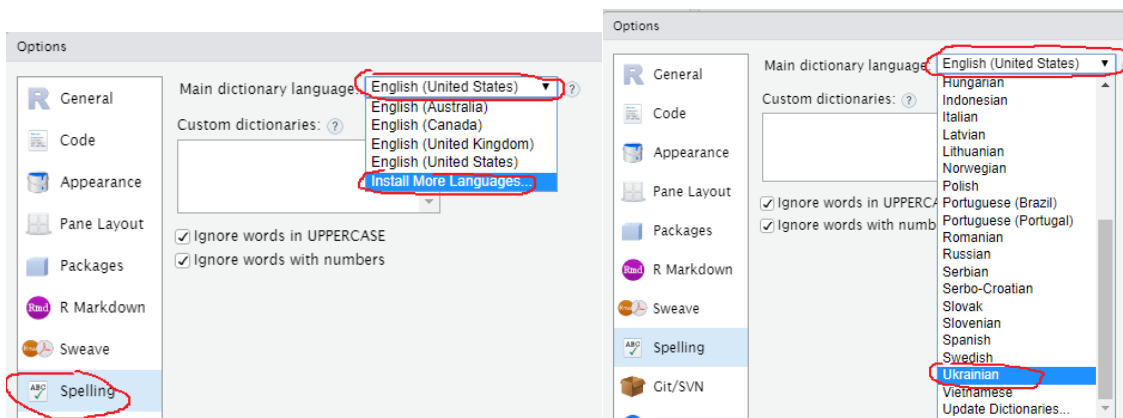
Спробуйте змінити розмір вікна редактора перетягуючи границю.



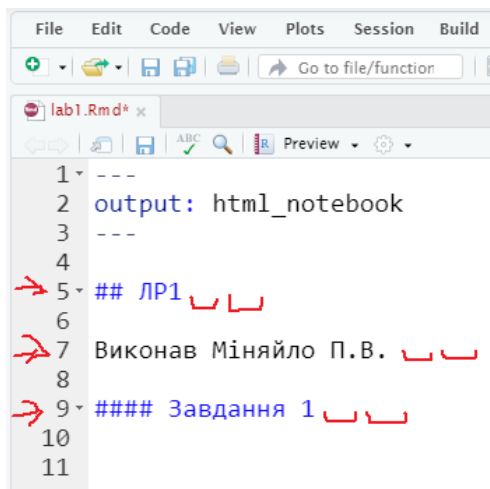
Збільшить розмір шрифта редактора (робиться один раз)



Додайте перевірку української орфографії (робиться один раз)

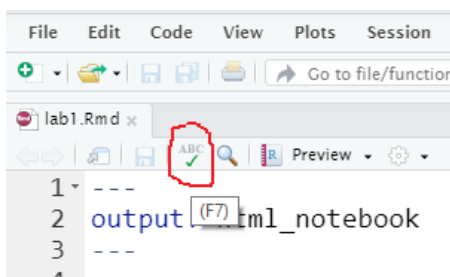


У вікні редактора записника lab1, в рядках 5-10, наберіть наступний текст. (В кінці тексту, в рядках 5, 7 та 9 додайте по **два пропуски**, після чого перехід на новий рядок!!!!)

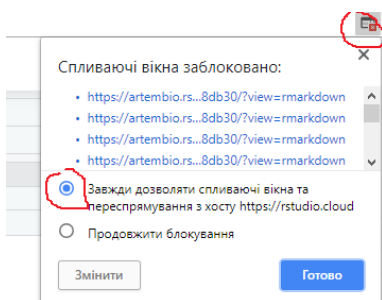


```
1 ---
2 output: html_notebook
3 ---
4
5 ## ЛР1
6
7 Виконав Міняйло П.В.
8
9 #### Завдання 1
10
11
```

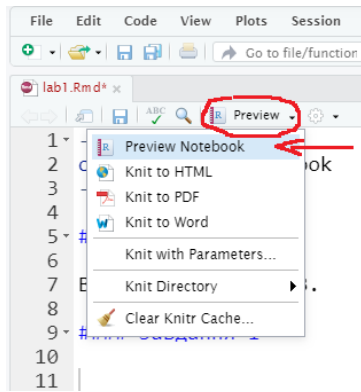
Запустити перевірку орфографії



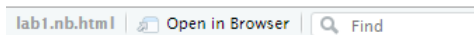
Якщо ви працюєте з сайтом rstudio.cloud, в браузері у вікні r-studio розблокуйте спливаючі повідомлення від цього сайту (робиться один раз)



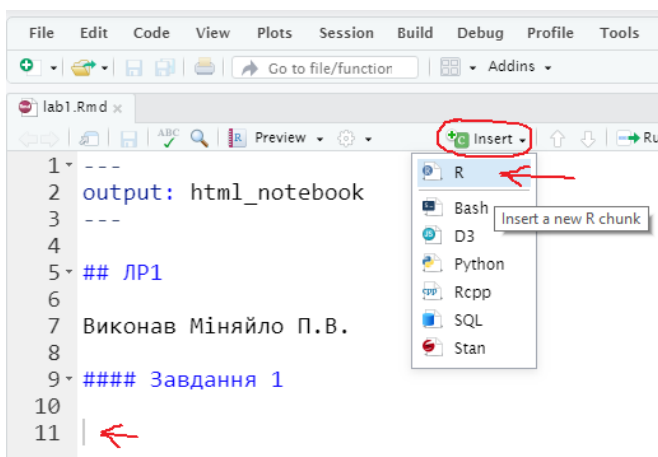
Запустити попередній перегляд записника



В новому вікні ви маєте побачити таке:

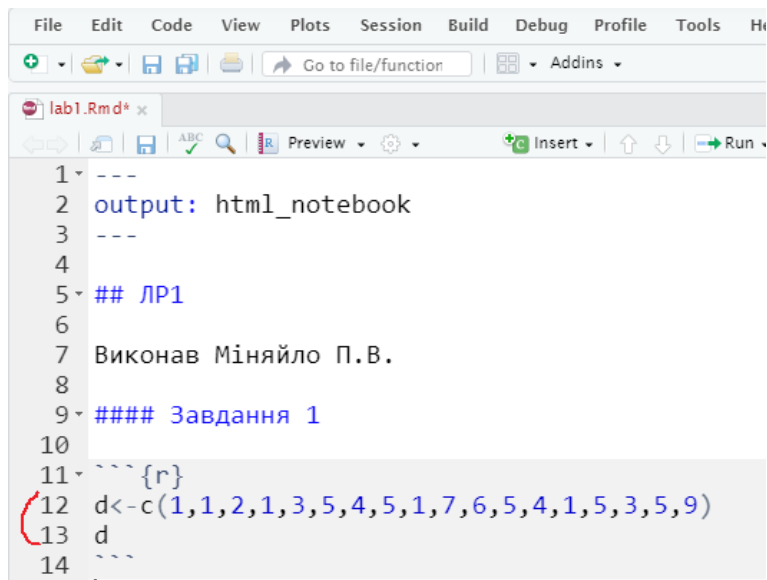


Поверніться до редактора, додайте в рядку 11 «кавалок коду» мовою R (R chunk)



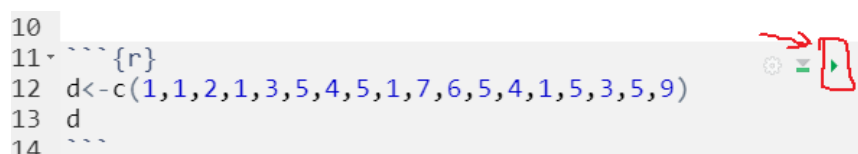
Зверніть увагу що блок коду починається рядком в якому знаходяться символи `{r}`, та закінчується рядком в якому знаходяться символи ````.

Напишіть такий код:



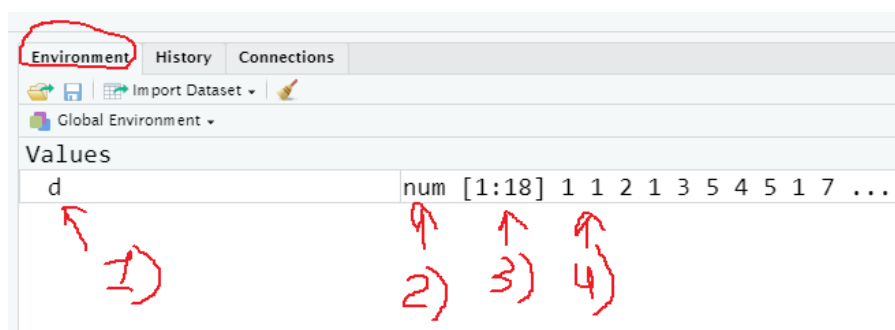
```
1 ---
2 output: html_notebook
3 ---
4
5 ## ЛР1
6
7 Виконав Мінняло П.В.
8
9 #### Завдання 1
10
11 {r}
12 d<-c(1,1,2,1,3,5,4,5,1,7,6,5,4,1,5,3,5,9)
13 d
14 {r}
```

Запустіть поточний кавалок з кодом:



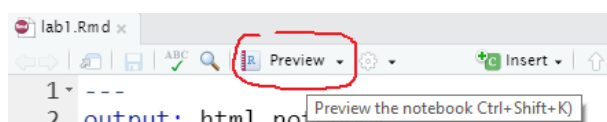
```
10
11 {r}
12 d<-c(1,1,2,1,3,5,4,5,1,7,6,5,4,1,5,3,5,9)
13 d
14 {r}
```

Подивіться як зміниться вміст вікна в Environment. В цьому вікні відображається список всіх змінних що знаходяться в пам'яті. В нашому випадку в пам'яті буде лишень одна змінна з ім'ям d (1). Це вектор у якого 18 елементів (3). Цей вектор є вектором типу num (2), тобто в його комірках знаходяться дійсні числа. Числа що розташовані в перших комірках ви також можете побачити в цьому вікні (4).



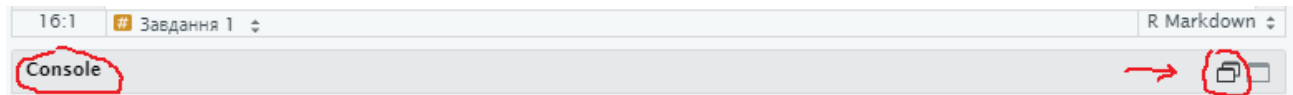
Values
d num [1:18] 1 1 2 1 3 5 4 5 1 7 ...

Запустіть попередній перегляд записника.



```
1 ---
2 output: html_notebook
```

Розгорніть вікно командного рядка (консоль, console).



Послідовно, по одній, введіть команди що наведені нижче. (Слідкуйте за результатами кожної з них. Ви маєте вміти пояснити що робить кожна з команд. Чому команда `c+1` видає помилку. Зверніть увагу на зміни у вікні `enviroment` після виконання команд `a<-c(3,10)` та `b<-a+3`)

`d`

`a<-c(3,10)`

`a+3`

`b<-a+3`

`c+1`

`diff(a)`

`mean(a)`

`sort(d)`

`cat('sss')`

`cat('sss',4,5)`

`cat('sss\n',4,5)`

`cat('sss\n',4,'\n',5)`

Під першим кавалком коду додайте ще один. Перед новим кавалком коротко напишіть значення його команд (**будуємо гістограму**). Цей текст носить пояснювальний характер Він коротко пояснює призначення коду.

```
11 данні
12 ~~~{r}
13 d<-c(1,1,2,1,3,5,4,5,1,7,6,5,4,1,5,3,5,9)
14 d
15 ~~~

[1] 1 1 2 1 3 5 4 5 1 7 6 5 4 1 5 3 5 9

16 будуємо гістограму
17 ~~~{r}
18 hist(d, main='кількість пальців на двох руках у слюсарів',
19      xlab='Кількість пальців', ylab='Частота')|
20 ~~~
```

Запустіть кавалок

Додайте ще один кавалок з кодом. Опишіть його призначення.

```

20
21 обчислюємо статистичні показники
22 ```{r}
23 cat('середнє:\n')
24 mean(d)
25 cat('медіана:\n')
26 median(d)
27
28 cat('діапазон:\n')
29 range(d)
30 diff(range(d))
31 cat('1ша та 2га квартиль:\n')
32 quantile(d,c(0.25,0.75))
33 cat('стандартне відхилення:\n')
34 sd(d)
35 ```

```

Запустіть кавалок.

Додайте ще один кавалок з кодом. Опишіть його призначення. Код в цьому кавалку буде робити те саме що і в попередньому. Відмінність в способі збереження обчислених результатів (вони будуть зберігатись в змінних з певними іменами які в подальшому можуть бути використані для додаткових обчислень або виведення на екран) та виведені результатів на екран

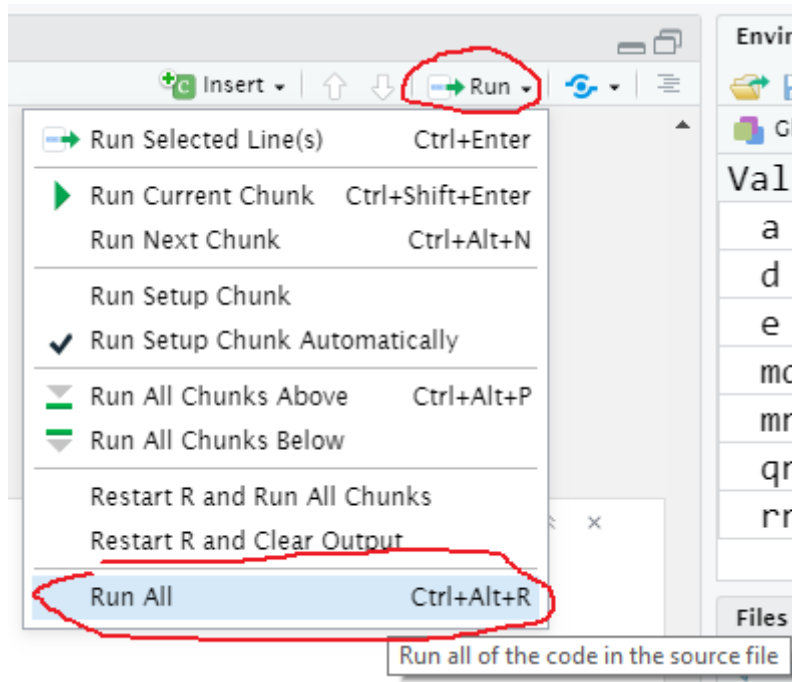
```

36
37 обчислюємо статистичні показники (варіант 2)
38 ```{r}
39 mn<-mean(d)
40 md<-median(d)
41 rng<-diff(range(d))
42 qnt<-quantile(d,c(0.25,0.75))
43 e<-sd(d)
44
45 cat('середнє:',mn,'\nмедіана:',md,'\ndіапазон:',rng,'\n1ша та 2га
46   квартиль:',qnt,'\nстандартне відхилення:',e)

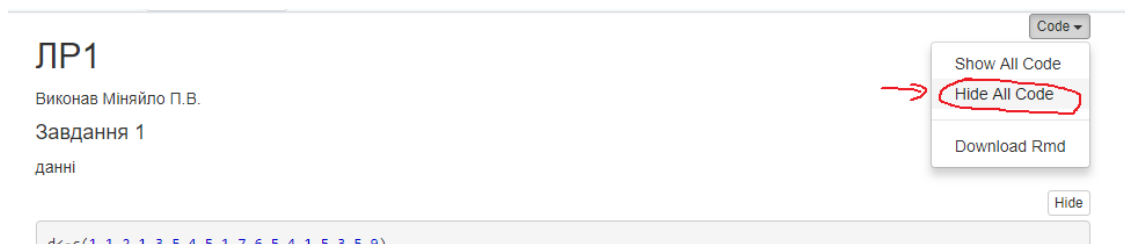
```

Запустіть повторно всі кавалки з кодом записника lab1 (це може знадобитись наприклад в тому випадку коли ви змінюєте дані що розташовані на початку вашого записника. Після чого ви хочете перерахувати всі результати.)





Включити попередній перегляд записника. Сховайте весь код, залишивши тільки текст та результати обчислень.



Після того як завдання буде показане викладачу, видаліть частину коду з рядків 37-46 (у вас нумерація рядків може відрізнятись від наведеної на малюнках вище)

## Доданок. Імпорт даних

Всі команди щоб будуть наведені в цьому додатку мають бути виконані вами в консолі програми R.

Часто статистичні дані представляються у вигляді таблиць. Ці таблиці можуть бути збережені у вигляді різноманітних файлів. Наприклад вони можуть зберігатись у вигляді текстових файлів (такі файли мають розширення .txt, .dat, .csv ...) або у вигляді файлів Excel (файли з розширенням .xls, .xlsx). Ми розглянемо яким чином ці таблиці можуть бути перенесені з цих файлів в пам'ять програми R та яким чином їх далі можна використовувати. В цьому прикладі в файл з таблицею ми створимо вручну. Для збереження таблиці ми будемо використовувати текстовий файл. Уявімо що в експерименті ми годували щурів двома різними дієтами (позначимо їх як дієта номер 1 та дієта номер 2). В кінці експерименту щури кожної з груп були зважені, частина результатів цього експерименту наведено в наступній таблиці:

N	дієта	вага
1	1	111
2	2	125
3	1	109
4	2	115

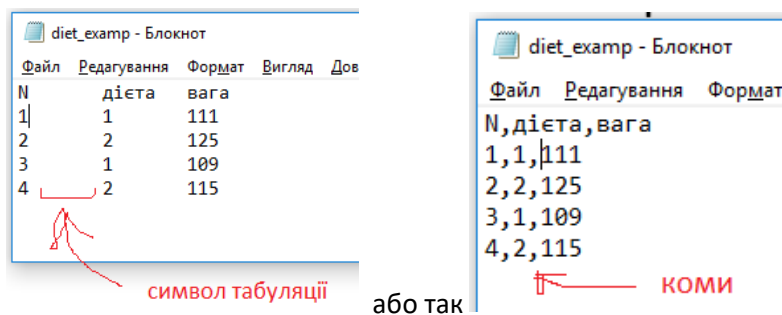
де стовпці:

**N** – номер щура

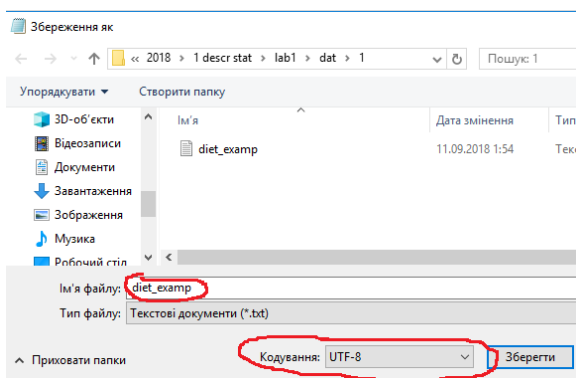
**дієта** – тип дієти

**вага** – вага щура в кінці експерименту

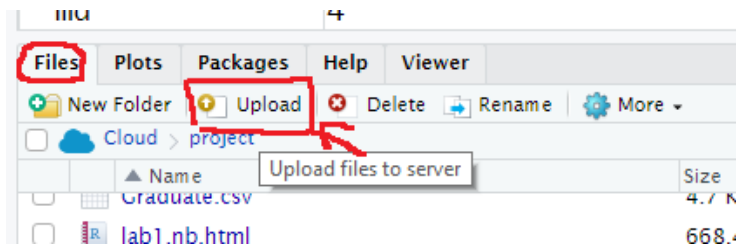
Перенесемо цю таблицю в текстовий файл. В блокноті створіть новий текстовий файл та наберіть в ньому нашу таблицю так:



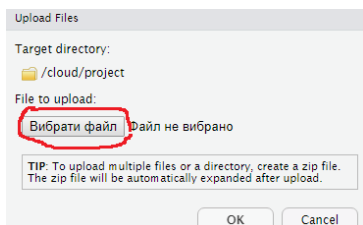
Збережіть файл під ім'ям diet\_examp (наприклад), кодування оберіть **UTF-8**:



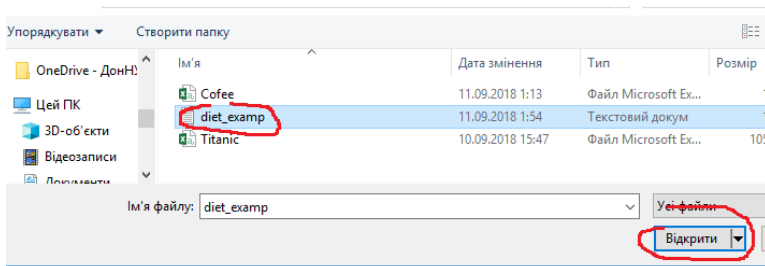
Файл треба зберегти в поточну теку програми R. Якщо ви працюєте на сайті RStudioCloud, цей файл з жорсткого диску (збережіть його де інде) ви маєте перенести в хмару. Натисніть



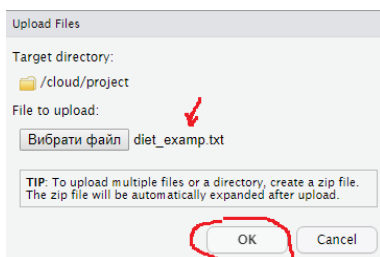
далі



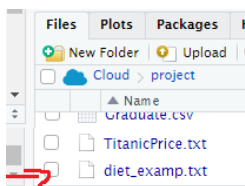
перейдіть в локальну теку в яку ви зберегли diet\_exam.txt:



далі

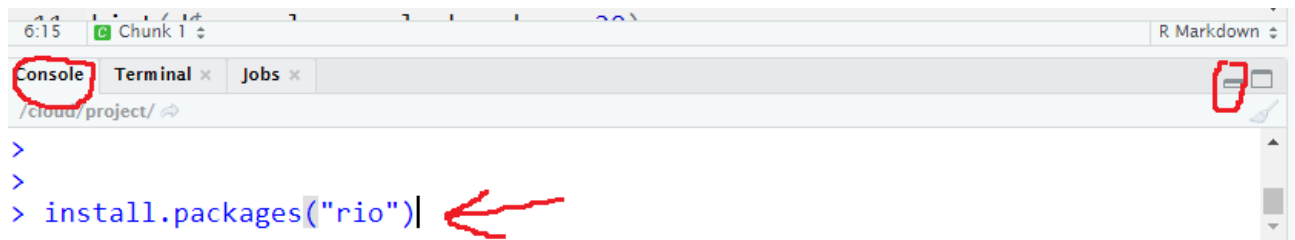


перевірте, файл має з'явитись в списку



Для того аби завантажити цю таблицю з нашого файлу diet\_exam.txt в пам'ять програми R ми будемо використовувати бібліотеку **rio** в якій для цього існує функція **import**. Перевага цієї бібліотеки полягає в тому що вона здатна імпортувати файли багатьох форматів використовуючи лишень одну команду **import**. Використання для цих цілей базових функцій R може бути дещо складніше тому для тих хто тільки починає використовувати програму R пакет функції бібліотеки **rio** може бути кращим варіантом. Ця бібліотека не входить до базової версії програми R тому її потрібно встановити в вашій системі (це робиться один раз, якщо ця бібліотека раніше не була встановлена вами). Для того аби встановити цю бібліотеку **в командному рядку програми R** введіть наступну команду:

```
install.packages("rio")
```



Після встановлення паркету в вашій системі з'явиться команда **import** за допомогою якої ви можете завантажувати дані з різноманітних файлів в пам'ять програми R. Кожен раз, після перезапуску програми, аби можна було використовувати функції цієї бібліотеки треба виконати команду (в командному рядку, або в вашому записнику):

```
library('rio')
```

Зазвичай такі команди розміщують на початку записника (в першому кавалку).

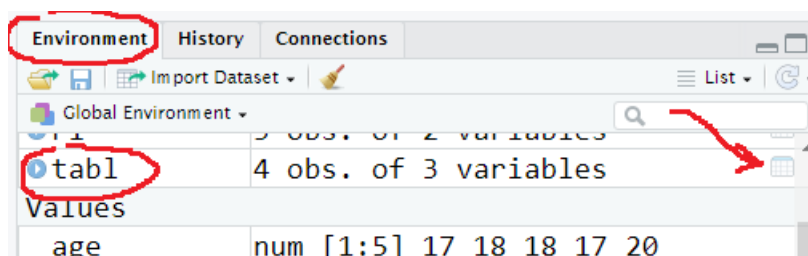
Далі завантажено таблицю з нашого текстового файлу та збережемо її в пам'яті програми в змінній з ім'ям **d**:

```
tabl<-import('diet_examp.txt')
```

Після виконання цієї команди давайте подивимося на вміст змінної **tabl**, що з'явилась в пам'яті. Для цього в командному рядку напишіть команду:

```
tabl
```

Ця команда буде відображати перші рядки вашої таблиці. Змінна **tabl** має спеціальний тип який називається **фрейм даних**. Ще один спосіб відобразити таблицю, клацніть мишкою тут



Розглянемо яким чином можна використовувати дані з завантаженої в пам'ять таблиці. Наприклад нам може знадобитись лише певний стовпчик з усієї таблиці (для побудови гістограми, обчислення середнього по числам з цього стовпчика і т.д.). Припустімо що ми збираємося знайти середню масу всіх щурів що приймали участь в нашому експерименті. Для цього нам зрозуміло потрібні дані з другого стовпчику таблиці, ці дані нам потрібно передати в команду **mean**. Для того аби виділити з усієї таблиці **tabl** лише один її стовпчик (**vara**) використовується така команда:

```
tabl$vara
```

(цю команду не набирайте вручну. Зробіть так, напишіть **tabl\$** та натисніть клавішу **Tab**, з випадаючого списку оберіть потрібну назву стовпця. Зауважимо, що для того аби це спрацювало, тобто з'явився список всіх назв стовпчиків, в пам'яті має існувати **tabl**, який перед цим ви завантажили через **tabl<-import('diet\_examp.txt')**)

Для того аби передати ці дані в команду він використовується така команда:

```
mean(tabl$vara)
```

Зауваження. Дата фрейм **tabl** це ціла таблиця, **tabl\$vara** окремий її стовпчик. Кожен стовпчик таблиці дата фрейм це вектор.

Можна скопіювати якийсь стовпчик в окремий вектор. Наприклад команда

**m<-tabl\$vara**

скопює всі значення стовпчика **vara** в вектор з ім'ям **m**. Виведіть вміст цього вектору на екран:

**m**

Маючи вектор **m**, команду **mean( tabl\$vara )** ми можемо виконати так

**mean( m )**

Завантажену в пам'ять таблицю можна модифікувати. Розглянемо як можна змінити всі значення якогось зі стовпчиків. Наприклад давайте в таблиці **tabl** переведемо грами в кілограми. Для цього нам кожне зі значень стовпчика **vara** необхідно розділити на 1000. Зробити це можна так:

**tabl\$vara <- tabl\$vara / 1000**

«Читається» ця команда так (з право на ліво): всі числа таблиці **tabl**, стовпчика **vara** ділимо на 1000; отриманий результат збережемо в стовпчику **vara (<-)** , замінивши попередні значення.

Для перевірки результату, виведіть таблицю на екран:

**tabl**

## Доданок. Як побудувати діаграму

Іноді, для категоріальних даних, доводиться створювати графік кількості кожного елементу у вигляді стовпчиків. Наприклад, маємо вектор віку 5 студентів першокурсників (всі наведені нижче команди запустить в командному рядку):

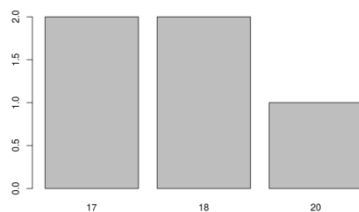
```
age <- c(17,17,18,18,20)
```

В R є вбудована команда `barplot` за допомогою якої ми можемо побудувати діаграму яка покаже скільки ми маємо першокурсників різного віку. Якщо просто виконати команду `barplot(age)` це не дасть бажаного результату. В такому вигляді команда створить діаграму з 10 стовпчиками, висота яких буде 17,17,18,18,20 (це значення з вектору `age`). Але ми хочемо знати кількість студентів у кожній віковій категорії. Таких категорій три: 17,18,20. В цих категоріях є 2, 2 та 1 студент відповідно. Це за нас легко може знайти функція `table()`:

```
table(age)
```

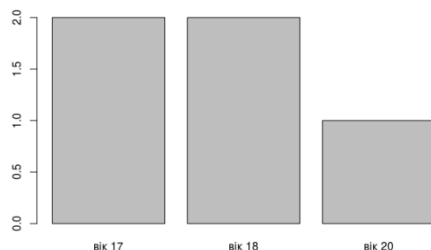
Саме результати цієї команди і потрібно передати в команду `barplot`:

```
barplot( table(age) )
```



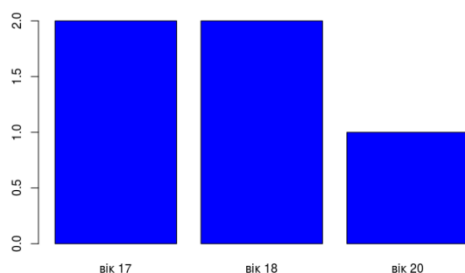
Якщо потрібно категоріям дати інші імена:

```
barplot( table(age), name=c('вік 17','вік 18','вік 20') )
```



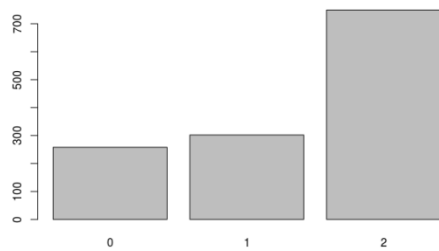
Колір ....

```
barplot( table(age), name=c('вік 17','вік 18','вік 20') , col='blue' )
```



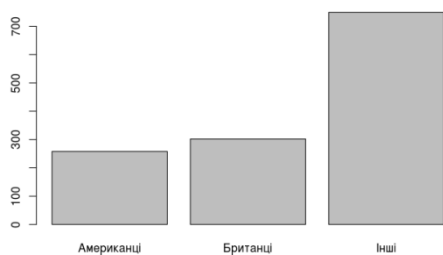
Наступний приклад стосується таблиці `Titanic.csv`. Побудемо діаграму кількості громадян різних країн на кораблі. Для цього використаємо стовпчик `d$Residence`, команди `table` та `barplot`:

```
barplot( table(d$Residence) )
```



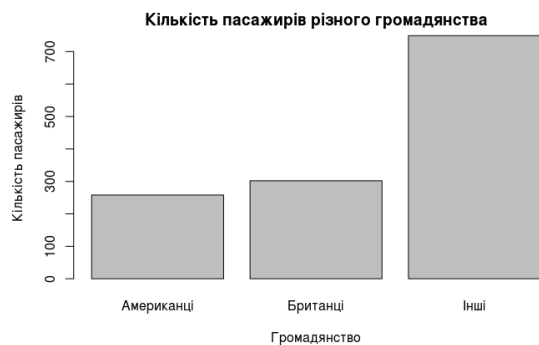
Підпишемо категорії:

```
barplot(table(d$Residence),names=c('Американці','Британці', 'Інші'))
```



Підпишемо також осі та заголовок:

```
barplot(table(d$Residence),names=c('Американці','Британці', 'Інші'),main='Кількість пасажирів різного громадянства', ylab='Кількість пасажирів', xlab='Громадянство')
```



## Доданок. Як з таблиці отримати дані тільки для певної категорії.

Наступний приклад стосується таблиці Titanic.csv. Припустімо що таблицю з файлу Titanic.csv ви завантажили в пам'ять R в фрейм даних що має ім'я **tabl**. Зі стовпчика ціна квитка (тобто **tabl\$fare**) нам потрібно отримати тільки ціну квитка тих пасажирів які загинули (для цих пасажирів в стовпчику **tabl\$survived** стоїть значення 0). Ці данні можна отримати так:

```
tabl$fare[ d$survived==0 ]
```

«прочитати» цю команду можна так: зі стовпчика **tabl\$fare** ми відбираємо тільки ті комірки які знаходяться в тих же рядках таблиці, в яких в стовпчику **d\$survived** стоять нулі. Візуально це можна представити так:

pclass	survived	...	fare	
3	0	...	7.55	
3	0	...	20.25	
3	0	...	20.25	
3	1	...	20.25	
3	1	...	7.65	
3	1	...	7.65	
2	0	...	24	
2	1	...	24	
3	1	...	7.925	
3	1	...	7.2292	
3	0	...	7.25	
3	0	...	8.05	
3	0	...	9.475	
...	...	...	...	

**tabl\$fare[ d\$survived==0 ]**

**tabl\$fare[ d\$survived==1 ]**

Ціни квитків пасажирів що вижили (для цих пасажирів в стовпчику **tabl\$survived** стоїть значення 1):

```
tabl$fare[ d$survived==1 ]
```

Таким чином якщо наприклад ви хочете побудувати гістограму розподілу ціни квитків пасажирів що вижили, треба написати:

```
hist( tabl$fare[ d$survived==1 ] )
```

Якщо хочете дізнатись скільки в середньому коштував квиток для пасажирів що загинули:

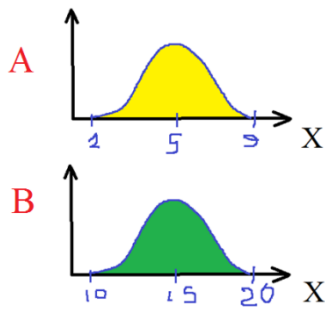
```
mean( tabl$fare[ d$survived==0 ] )
```

І так ділі ...

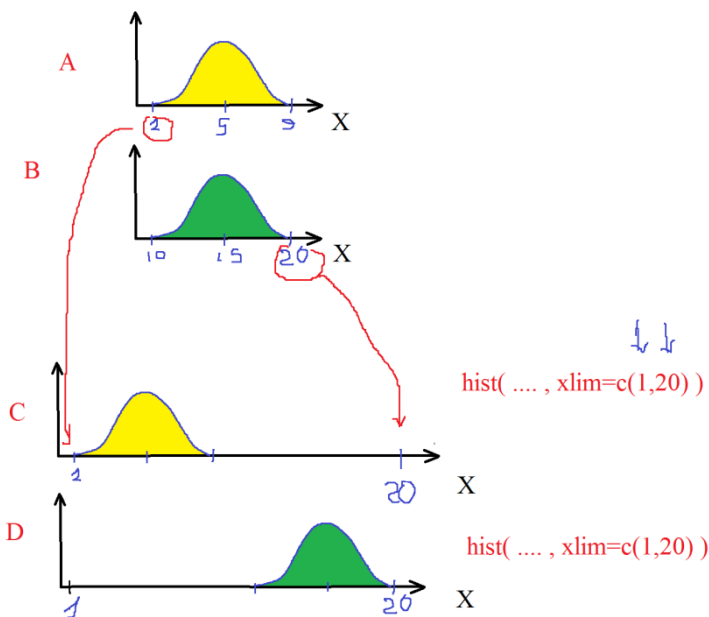


## Доданок. Гістограма як задати масштаб осей

Іноді виникає необхідність змінити масштаб осі абсцис гістограми. Розглянемо приклад. Припустімо ми маємо дві гістограми які ми хочемо порівняти між собою.



Як ми можемо бачити з малюнка для гістограми **A**, середнє розташоване приблизно точці  $x = 5$ . Для гістограми **B** середнє приблизно знаходиться в точці  $x = 15$ . Але візуально гістограми виглядають однаковими (якщо не аналізувати діапазони в яких лежать ці дві гістограми по осі абсцис). Це є наслідком того що ми маємо різний масштаб по осі абсцис на цих двох гістограмах. Границі осі абсцис на гістограмі **A** лежать в діапазоні від 1 до 9. Тоді як на другій від 10 до 20. Такий масштаб обирається автоматично командою `hist` при побудові кожної з гістограм. Для того аби цю відмінність представити більше наглядно нам вручну потрібно змінити масштаб по ось абсцис, на обох гістограмах його треба зробити однаковим. Новий масштаб буде від 1 до 20:



Як це зробити практично. Спочатку в записнику побудуйте дві гістограми не вказуючи масштаб по осі абсцис:

```
hist(...)
```

```
hist(...)
```

Далі дивлячись на побудовані гістограми визначте яким має бути спільний масштаб. Після чого модифікуйте команди наступним чином:

```
hist(..., xlim=(1,20) )
```

```
hist(..., xlim=(1,20) )
```

(замість 1 та 20 впишіть свої границі).

## Доданок Відсоток вижилих серед чоловіків та жінок на Титаніку

Для набору даних Титанік побудуємо діаграму яка буде показувати який після катастрофи залишився відсоток чоловіків що виїжджали а також жінок . Ми почнемо з кінця припустімо що серед чоловіків вижило 40% а серед жінок вижило 60%( далі Ми змінимо ці цифри врахувавши їх з реальних даних). В цьому випадку діаграма буде виглядати приблизно наступним чином .



На цьому ж малюнку показана команда за допомогою якої можна побудувати таку діаграму. Як і раніше для побудови діаграми ми будемо використовувати вбудовану графічну функцію `barplot`. Перший аргумент цієї функції `c(40, 60)` це вектор в якому ми задаємо висоти всіх стовпців які будуть на нашій діаграмі (у нас буде два стовпця). Другий аргумент `name=c('Чоловіки','Жінки')` це назви цих стовпців, які будуть написані по осі абсцис (кількість назв має співпадати з кількістю стовпців, тобто з кількістю чисел які ви напишете в першому аргументі). Третій аргумент `col=c('red','green')` це вектор який задає колір кожного стовпчика ( знову ж таки в цьому векторі має бути стільки ж назви кольорів скільки на діаграмі буде стовпчиків). Якщо присвоїти аргумент `col` лишень один колір, всі стовпчики будуть пофарбовані цим кольором (наприклад так `col='red'`). Четвертий аргумент `ylab` визначає підпис осі ординат. Останній аргумент `main` створить заголовок для нашої діаграми.

Для того аби побудувати діаграму по реальних даних в наведеній вище, команді достатньо замінити два числа 40 та 60, на реальні відсотки. Розглянемо яким чином це можна зробити. Припустімо що таблицю з файлу `Titanic.csv` ви завантажили в пам'ять R, в фрейм даних що має ім'я `tabl`. Зі стовпчика

ціна квитка (тобто **tabl\$fare**) нам потрібно отримати тільки ціну квитка тих пасажирів які загинули (для цих пасажирів в стовпчику **tabl\$survived** стоїть значення 0). Ці данні можна отримати так:

```
tabl$fare[ d$survived==0 ]
```

Для того аби побудувати діаграму по реальних даних в наведеній вище команді **barplot(...)** достатньо замінити два числа 40 та 60 на реальні відсотки. Розглянемо яким чином можна обрахувати реальні значення. Для того аби порахувати відсоток чоловіків що вижили спочатку скопіюємо з нашої таблиці **tabl** в окремий вектор дані по живучості чоловіків . Це можна зробити наступною командою (ці данні ми скопіюємо в вектор з ім'ям **sman**)

```
sman <- tabl$survived[ tabl$Gender==0 ]
```

Давайте прочитаємо цю команду. Якщо б було написано так **sman <- tabl\$survived**, ми прочитали би це як «скопіювати в вектор **sman** стовпчик **survived** таблиці **tabl**». Якщо ж написано як у нас **sman <- tabl\$survived[ tabl\$Gender == 0 ]**, це означає що зі стовпчика **survived** (живучість) ми скопіюємо не всі данні, а тільки ті для яких в стовпчику **Gender** (стать) стоїть значення **0**. Стать «чоловік» позначається в стовпчику **Gender** як **0**:

таблиця tabl в пам'яті R					
		survived	Residence	...	Gender
невижив	→	0	2	...	0 ← чоловік
		0	2	...	1
		0	2	...	1
вижив	→	1	2	...	0 ← чоловік
вижив	→	1	2	...	0 ← чоловік
		0	0	...	1
		0	0	...	1
		...	...	...	...

копіюємо  
в вектор  
**sman**

ті комірки в стовпчику **survived** для яких в стовпчику **Gender** стоїть 1  
**sman <- tabl\$survived[ tabl\$Gender==0 ]**

Оператор **==** це логічне дорівнює. Оце **[ tabl\$Gender==0 ]** знайде номери комірок в стовпчику **Gender** в яких стоять нулі.

Таким чином відсоток тих що вижили буде дорівнювати:

$(\text{кількість одиниць в векторі sman}) / (\text{кількість всіх елементів в векторі sman}) * 100$

Для того аби порахувати кількість всіх елементів в якомусь векторі слугує команда **length**. Наприклад якщо в функцію **length** передати вектор з двома елементами **length( c(40,60) )** вона в якості результату поверне нам число 2. Кількість елементів в векторі **sman** буде дорівнювати:

**length( sman )**

Введіть цю команду в командному рядку, подивіться результат. Для того аби знайти кількість одиниць (кількість чоловіків що вижили) нам просто необхідно знайти суму всіх елементів вектора

**sman**. Це можна зробити командою **sum**. Наприклад `sum( c(1,0,1) )` видасть результат  $2=1+0+1$ . Подивіться, запустивши в консолі команду

```
sum( sman )
```

скільки чоловіків вижило на Титаніку.

Таким чином відсоток чоловіків що вижили буде дорівнювати:

```
sman_num_survived <- sum(sman) / length(sman) * 100
```

Цей відсоток ми зберегли в змінній з ім'ям **sman\_num\_survived** ( для того аби потім не переписувати це число вручну в команду `barplot`). Подивіться на це число написавши в консолі `sman_num_survived`. Аналогічним чином ми можемо порахувати відсоток жінок що вижили:

```
swomen <- tbl$survived[ tbl$Gender==1 ]
```

```
swomen_num_survived <- sum(swomen) / length(swomen) * 100
```

Ну і нарешті побудемо діаграму по реальних даних

```
barplot( c(sman_num_survived, swomen_num_survived), name=c('Чоловіки','Жінки') ,  
col=c('red','green') , ylab='Кількість вижилих, %', main='Відсоток вижилих серед чоловіків та жінок на  
Титаніку' )
```

Зберемо до купи всі команди що знадобляться нам для побудови діаграми в записнику:

```
sman <- tbl$survived[ tbl$Gender==0 ]
```

```
sman_num_survived <- sum(sman) / length(sman) * 100
```

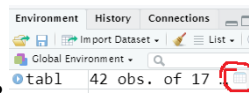
```
swomen <- tbl$survived[ tbl$Gender==1 ]
```

```
swomen_num_survived <- sum(swomen) / length(swomen) * 100
```

```
barplot( c(sman_num_survived, swomen_num_survived), name=c('Чоловіки','Жінки') , col=c('red','green') ,  
ylab='Кількість вижилих, %', main='Відсоток вижилих серед чоловіків та жінок на Титаніку' )
```

## Доданок. Що робити якщо відсутня частина даних?

Описане тут стосується набору даних «Титанік». Припустімо що таблицю з файлу Titanic.csv ви завантажили в пам'ять R, в фрейм даних що має ім'я **tabl**.



Якщо ви подивитись на вміст таблиці **tabl** (клікніть **tabl** 42 obs. of 17) ви можете побачити що в деяких стовпчиках замість значень, записане слово **NA** (Not available – не доступне). Так наприклад стовпчику **fare** в рядку 1156 замість ціни квитка стоїть значення NA. Тобто для деяких пасажирів були відсутні ми дані про те якою була ціна придбаного ними квитка, і вони не були внесені в таблицю (замість них в таблиці стоїть порожня комірка):

	pclass	survived	Residence	Name	...	fare
1154	2	0	1	Stokes, Mr. Philip Joseph	...	10.5
1156	3	0	1	Storey, Mr. Thomas	...	NA
1157	3	0	2	Stoytcheff, Mr. Ilia	...	7.89
1158	3	0	2	Strandberg, Miss. Ida Sofia	...	9.83

Відсутність значень буде впливати на статистичні обчислення які ми проводимо з даними . Наприклад давайте спробуємо знайти середню ціну квитка. Для цього в команду **mean** нам необхідно передати всі числа стовпчика **fare**:

```
mean( tabl$fare )
```

Однак в результаті виконання команди, замість середньої вартості квитка ми отримаємо значення NA (перевірте в консолі). Завдяки тому, що в цьому стовпчику деякі комірки «порожні» (мають значення NA) команда mean не може порахувати середнє.

Як виправити цю ситуацію? Ми можемо порахувати середнє по тих даних які є присутніми в цьому стовпчику але для цього необхідно сказати команді mean аби для обчислення середнього вона використовувала тільки наявні дані а порожні комірки пропускала . Це можна задавши додатковий аргумент:

```
mean( tabl$fare, na.rm = T)
```

Аргумент **na.rm** розшифровується як **remove NA** тобто усунути відсутні значення з даних. Цей аргумент приймає два значення T (true, істинно, так видаляти), або F (false, хибно, ні не видаляти). Аргумент **na.rm** приймають і багато інших команд, наприклад такі

**median**

**sd**

**quantile**

**var**

**range**

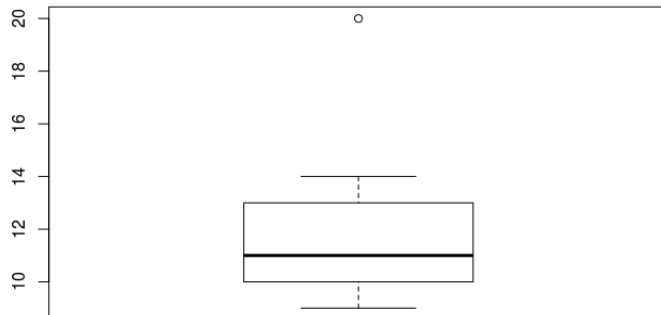
**sum**

.....

## Доданок. «Ящик з вусами»

```
d<-c(10,11,12,10,14,9,20)
```

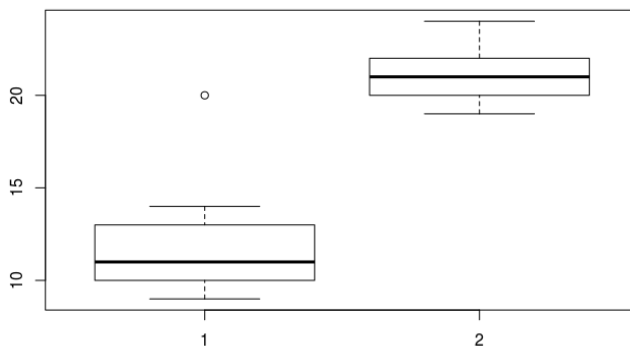
```
boxplot( d )
```



```
d1<-c(10,11,12,10,14,9,20)
```

```
d2<-c(20,21,22,20,24,19,22)
```

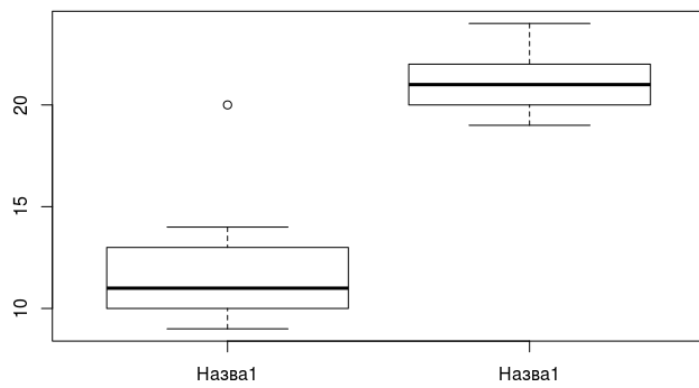
```
boxplot( d1, d2 )
```



```
d1<-c(10,11,12,10,14,9,20)
```

```
d2<-c(20,21,22,20,24,19,22)
```

```
boxplot( d1, d2 , names=c('Назва1','Назва1') )
```



```
d1<-c(10,11,12,10,14,9,20)
```

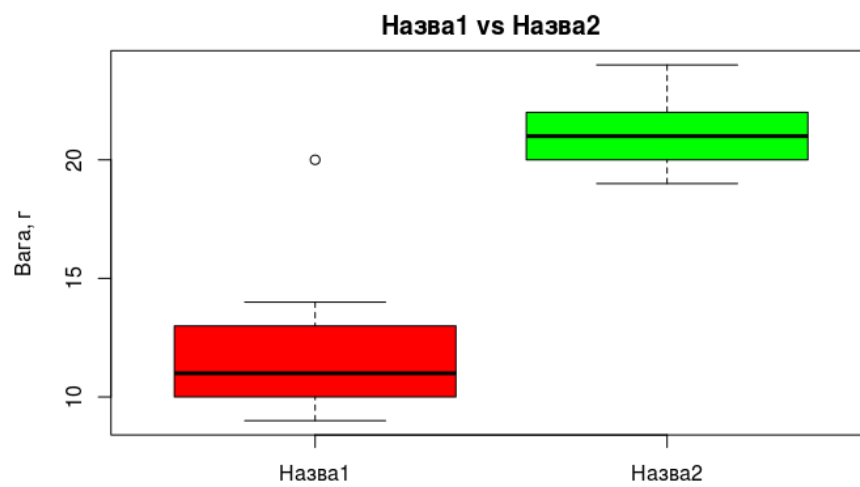
```
d2<-c(20,21,22,20,24,19,22)
```

```
boxplot(d1,d2,
```

```
names=c('Назва1','Назва2'),
```

```
main='Назва1 vs Название2' , ylab='Bara, r',
```

```
col=c('red','green'))
```

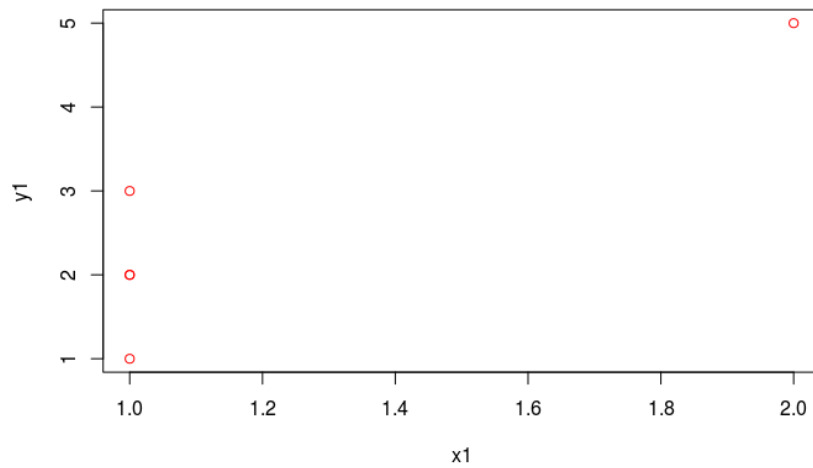


## Точкові діаграми

```
x1=c(1,1,2,1,1)
```

```
y1=c(2,3,5,2,1)
```

```
plot(x1,y1,xlab='x1', ylab = 'y1',col='red')
```



```
x1=c(1,1,2,1,1)
```

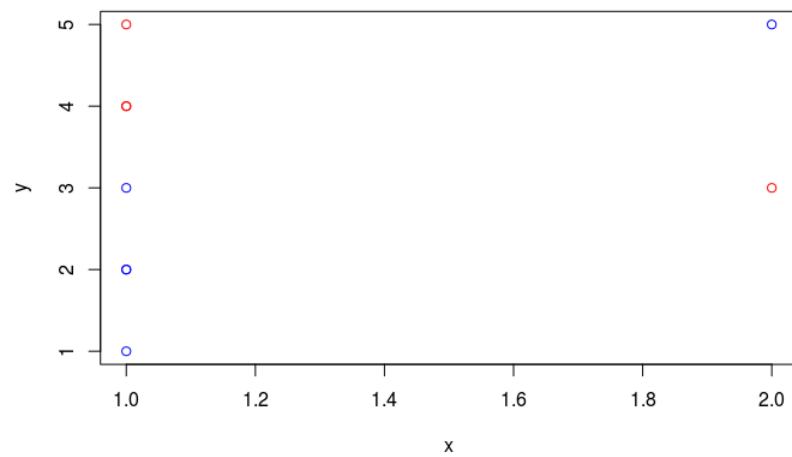
```
y1=c(2,3,5,2,1)
```

```
x2=c(1,1,2,1,1)
```

```
y2=c(4,5,3,4,6)
```

```
plot(x1,y1,col = "blue", xlab='x', ylab='y')
```

```
points(x2,y2,col = "red")
```





```
x1=c(1,1,2,1,1)
```

```
y1=c(2,3,5,2,1)
```

```
x2=c(1,1,2,1,1)
```

```
y2=c(4,5,3,4,6)
```

```
plot(x1,y1,col = "blue", xlab='x', ylab='')
```

```
points(x2,y2,col = "red")
```

```
legend(1.2,4, pch = 'o', legend=c("y1", "y2"), col=c("blue", "red") )
```

