Aleksander Jakóbczyk i Kacper Pasterniak

Sprawozdanie 1

Lista 1

Zad 1

Detergent

```
Detergent.df <- data.frame(Detergent)
Detergent.df %>% group_by(Temperature) %>% summarise(n = sum(Freq))
```

```
## # A tibble: 2 x 2
##   Temperature     n
##   <fct>       <dbl>
## 1 High          369
## 2 Low           639
```

```
Detergent.df %>% filter(Water_softness == "Soft") %>% group_by(Temperature) %>% summar
```

```
## # A tibble: 2 x 2
##   Temperature     n
##   <fct>       <dbl>
## 1 High          104
## 2 Low           222
```

```
Detergent.df %>% filter(Water_softness == "Medium") %>% group_by(Temperature) %>% summ
```

```
## # A tibble: 2 x 2
##   Temperature     n
##   <fct>       <dbl>
## 1 High          126
## 2 Low           218
```

```
Detergent.df %>% filter(Water_softness == "Hard") %>% group_by(Temperature) %>% summar
```

```
## # A tibble: 2 x 2
##   Temperature     n
##   <fct>       <dbl>
## 1 High          139
## 2 Low           199
```

Preference

```
# Detergent.df %>% group_by(Preference) %>% summarise(n = sum(Freq))
Detergent.df %>% filter(Water_softness == "Soft") %>% group_by(Preference) %>% summari
```

```
## # A tibble: 2 x 2
##   Preference     n
##   <fct>      <dbl>
## 1 Brand X      168
## 2 Brand M      158
```

```
Detergent.df %>% filter(Water_softness == "Medium") %>% group_by(Preference) %>% summa
```

```
## # A tibble: 2 x 2
##   Preference     n
##   <fct>      <dbl>
## 1 Brand X      169
## 2 Brand M      175
```

```
Detergent.df %>% filter(Water_softness == "Hard") %>% group_by(Preference) %>% summari
```

```
## # A tibble: 2 x 2
##   Preference     n
##   <fct>      <dbl>
## 1 Brand X      171
## 2 Brand M      167
```

Zad 2

```
ftable(Detergent, col.vars = "Temperature", row.vars = "Water_softness")
```

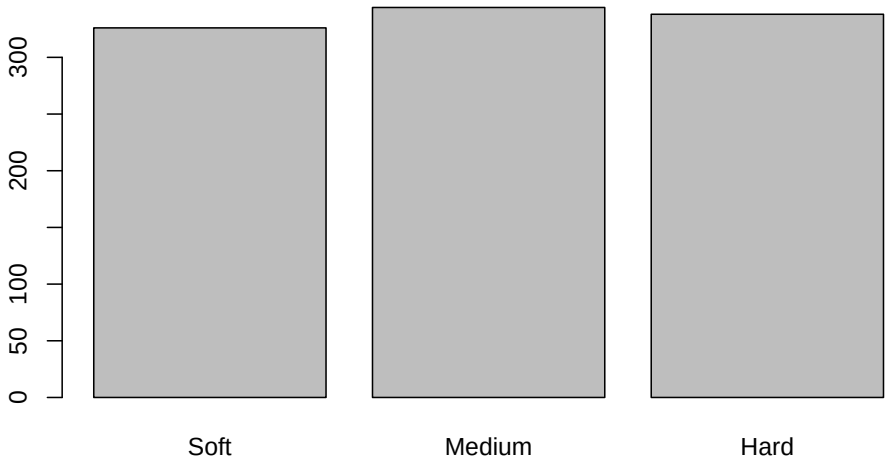
```
##                Temperature High Low
## Water_softness
## Soft                         104 222
## Medium                       126 218
## Hard                         139 199
```

```
structable(Temperature ~ Water_softness, Detergent) %>% addmargins()
```

```
##               Temperature
## Water_softness High Low  Sum
##         Soft    104 222  326
##         Medium  126 218  344
##         Hard    139 199  338
##         Sum     369 639 1008
```

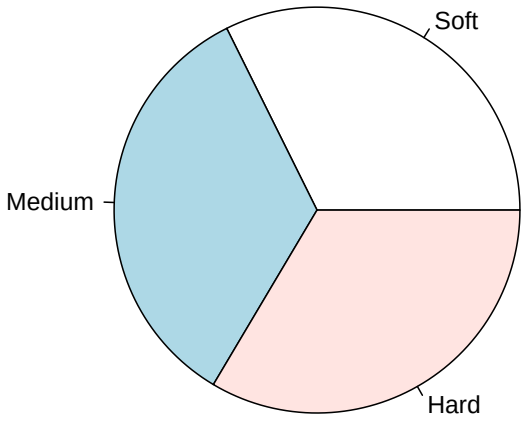
Zad 3

```
A <- apply(Detergent, "Water_softness", sum)
barplot(A)
```



Rysunek 1. Wykresy słupkowy dla zmniennej Water Softness.

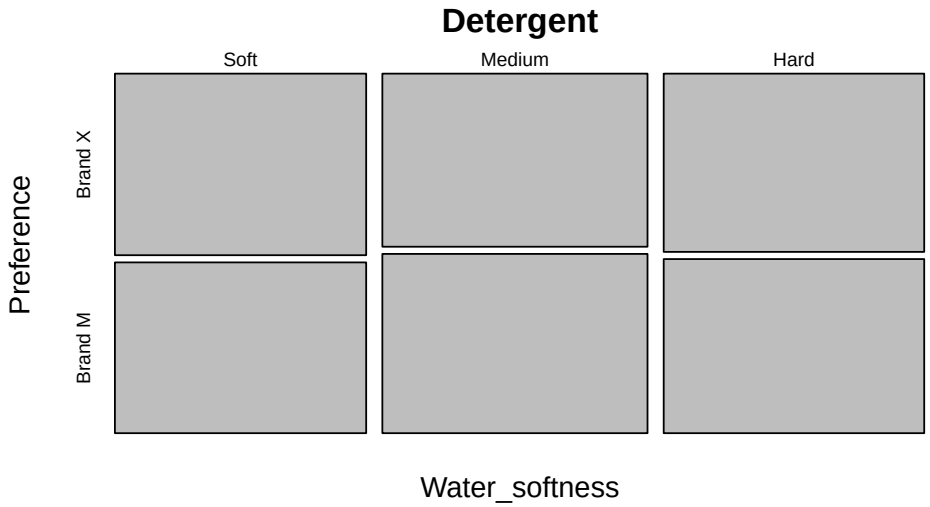
```
par(mar = c(2, 2, 2, 2))
pie(A)
```



Rysunek 2. Wykresy kołowy dla mniennej Water Softness.

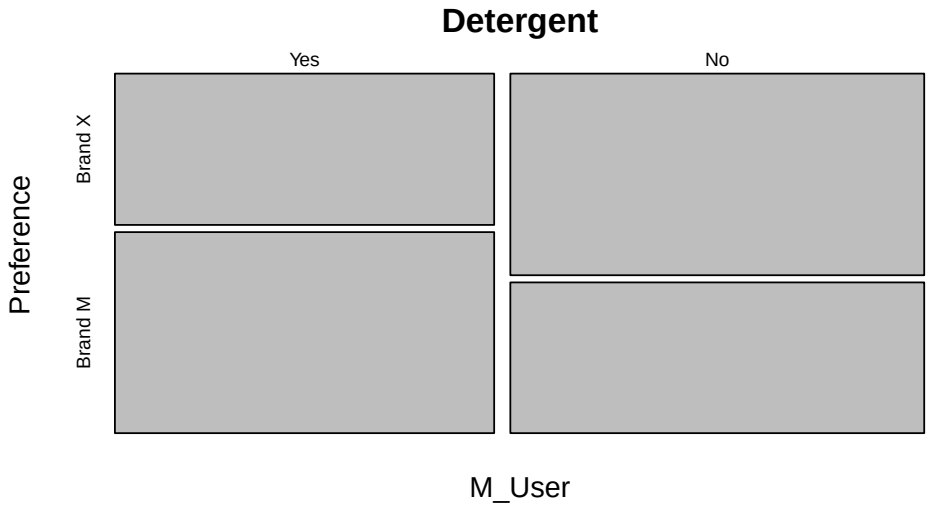
Zad 4

```
par(mar = c(2, 2, 2, 2))
mosaicplot(~Water_softness+Preference, data = Detergent)
```



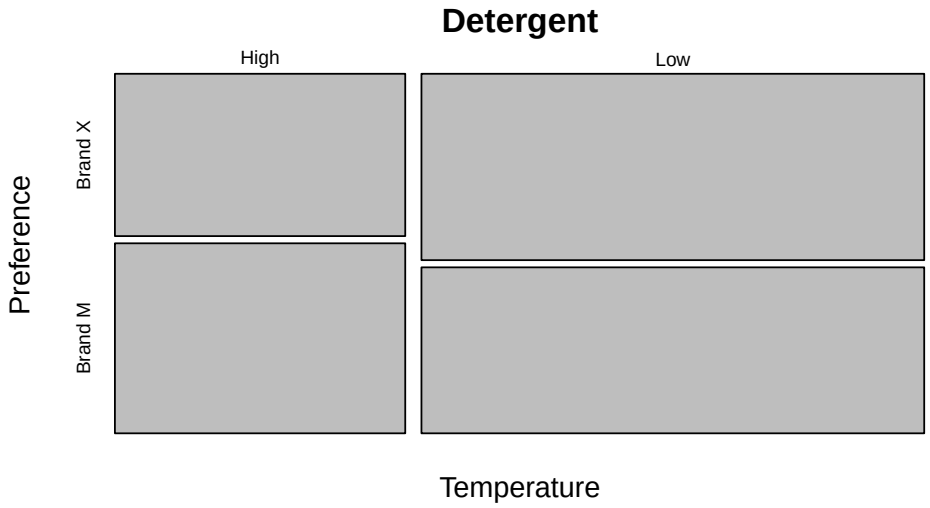
Rysunek 3. Wykres mozajkowy dla Preference i Water softness

```
par(mar = c(2, 2, 2, 2))
mosaicplot(~M_User+Preference, data = Detergent)
```



Rysunek 4. Wykres mozajkowy dla Preference i M User.

```
par(mar = c(2, 2, 2, 2))
mosaicplot(~Temperature+Preference, data = Detergent)
```



Rysunek 5. Wykres mozajkowy dla Preference i Temperature.

Lista 2

Zad 1

Losowanie ze zwracaniem:

```
ind <- sample(x=nrow(mtcars),size=nrow(mtcars)/10,replace=TRUE)
```

Wylosowane indeksy:

```
ind
```

```
## [1] 20 21 18
```

Wylosowane elementy z bazy danych:

```
mtcars[ind,]
```

```
##                 mpg cyl  disp hp drat    wt  qsec vs am gear carb
## Toyota Corolla 33.9   4  71.1 65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona  21.5   4 120.1 97 3.70 2.465 20.01  1  0    3    1
## Fiat 128       32.4   4  78.7 66 4.08 2.200 19.47  1  1    4    1
```

Losowanie bez zwracania:

```
ind <- sample(x=nrow(mtcars),size=nrow(mtcars)/10,replace=FALSE)
```

Wylosowane indeksy:

```
ind
```

```
## [1] 16 10 27
```

Wylosowane elementy z bazy danych:

```
mtcars[ind,]
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
```

Zad 2

Propozycja algorytmu:
1.Generujemy wektor zer o rozmiarze n
2.Dla każdego elementu tego wektora losujemy u z rozkładu jednostajnego U(0,1), jeśli u < p to dodajemy 1 do tego elementu
3.Krok 2 powtarzamy N razy

Algorytm opisany za pomocą funkcji w R:

```
bin <- function(n,p,N){
  y <- rep(0,n)
  for(j in 1:N){
    for(i in 1:n){
      u <- runif(1)
      if(u < p){
        y[i] <- 1 + y[i]
      }else{
        y[i] <-0 + y[i]
      }
    }
  }
  return(y)
}
```

gdzie: n - rozmiar próby, p - prawdopodobieństwo, N - ilość wywołań
Przykładowe użycie:

```
bin(10,0.4,5)
```

```
##  [1] 3 1 0 1 2 2 1 2 1 2
```

Sprawdzenie poprawności:
Wiemy, że średnia z takie próby powinna wynosić: np, a wariancja: np(1-p)

```
test <- bin(1000,0.4,10000)
mean(test)
```

```
## [1] 4002.444
```

```
var(test)
```

```
## [1] 2361.346
```

Wartości teoretyczne średniej i wariancji dla takich parametrów powinny wynosić kolejno 4000 i 2400. Nasze wyniki są bardzo bliskie co wskazuje na poprawność metody.