Hoja 1 de problemas y prácticas con R

Estadística Computacional I. Grado en Estadística

Departamento de Estadística e Investigación Operativa. Universidad de Sevilla

1. Crear un vector donde se repitan los códigos provinciales de Andalucía: 10 veces Almería, 10 veces Cádiz, . . . , 10 veces Jaén, 15 para Málaga y 18 Sevilla. Permutar aleatoriamente los elementos de dicho vector y calcular la tabla de frecuencias.

```
codigos=c(4,11,14,18,21,23,29,41)
#acceder a elementos:
codigorepetidos=rep(codigos,c(10,10,10,10,10,10,15,18))
codigorepetidos
```

```
##  [1]  4  4  4  4  4  4  4  4  4  4 11 11 11 11 11 11 11 11 11 11 14 14 14 14 14
## [26] 14 14 14 14 14 18 18 18 18 18 18 18 18 18 18 21 21 21 21 21 21 21 21 21 21
## [51] 23 23 23 23 23 23 23 23 23 23 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29
## [76] 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
```

Permuto aleatoriamente los elementes de dicho vector %sample me los permuta %Para calcular la tabla de frecuencias table %si queremos evitar que la permutacion varie voy a usar una semilla

```
set.seed(1)
codigospermutados=sample(codigorepetidos)
codigospermutados
```

```
##  [1] 29 18  4 18 41 21 11 41 23 23 14 23 29  4 29 18 41 41 21 18 18 29 21 18 11
## [26] 14 29 29 18 29 14 23 23  4 23 41 41 29 23 14 18 41  4 21 11 14 23 29 41 29
## [51] 29 14 41 29 41 11 11 18 11 29 41 29 21 29 41 41 21 11 11 21 41 41 21 41  4
## [76] 14 41 21 18  4 21 14  4 23 41 14  4 11 14 11  4 23  4
```

```
table(codigospermutados)
```

```
## codigospermutados
##  4 11 14 18 21 23 29 41
## 10 10 10 10 10 10 15 18
```

```
set.seed(1)
codigorepetidos %>%
  sample() %>%
  table()
```

```
## .
##  4 11 14 18 21 23 29 41
## 10 10 10 10 10 10 15 18
```

2. Con la ayuda de paste, crear un vector de nombres "Caso_1",. . . ,"Caso_30".

```
paste("Caso_",1:30,sep="") #equivalente a paste0("Caso_",1:30)
```

```
##  [1] "Caso_1"  "Caso_2"  "Caso_3"  "Caso_4"  "Caso_5"  "Caso_6"  "Caso_7"
##  [8] "Caso_8"  "Caso_9"  "Caso_10" "Caso_11" "Caso_12" "Caso_13" "Caso_14"
```

```
## [15] "Caso_15" "Caso_16" "Caso_17" "Caso_18" "Caso_19" "Caso_20" "Caso_21"
## [22] "Caso_22" "Caso_23" "Caso_24" "Caso_25" "Caso_26" "Caso_27" "Caso_28"
## [29] "Caso_29" "Caso_30"
```

```
#paste("Caso",1:30,c("a","b"),sep="_")
```

3. Generar dos vectores de tamaño 250, seleccionando aleatoriamente números enteros entre 0 y 999, sean x e y los vectores resultantes.
   i) Visualizarlos en dos columnas.
   ii) Construir el vector $y_2 - x_1, \ldots, y_{250} - x_{249}$.
   iii) Generar el vector $y_2 - y_1, \ldots, y_{250} - y_{249}$.
   iv) Construir el vector $x_1 + 2x_2 - x_3, x_2 + 2x_3 - x_4, ..., x_{248} + 2x_{249} - x_{250}$.
   v) Calcular la suma de los valores $1/(x_i + y_i)$.

```
set.seed(1357) #semilla del generador
#Vamos a generar los vectores aleatorimaente de tamaño 250, vamos a hacerlo con reemplazamiento
n=250;
x=sample(0:999,size=n, replace=T)
y=sample(0:999,size=n, replace=T)
#i)visualizarlo en columna
xy=cbind(x,y)
head(xy,10)
```

```
##         x   y
##  [1,] 139 894
##  [2,] 776 235
##  [3,] 537 316
##  [4,] 262 963
##  [5,] 451  69
##  [6,]  89  99
##  [7,] 272  70
##  [8,] 644 731
##  [9,]  81 848
## [10,] 955 220
```

Otro modo:(tidyverse)

```
set.seed(1357) #directamente digo lo que valen las columnas
n=250;
xyt=tibble(
x=sample(0:999,size=n, replace=T),
y=sample(0:999,size=n, replace=T))
xyt
```

```
## # A tibble: 250 x 2
##        x     y
##    <int> <int>
##  1   139   894
##  2   776   235
##  3   537   316
##  4   262   963
##  5   451    69
##  6    89    99
##  7   272    70
##  8   644   731
##  9    81   848
## 10   955   220
```

```
## # ... with 240 more rows
```

```
#ii)    Construir el vector $y_2-x_1,\ldots,y_{250}-x_{249}$.
v2=y[2:n] -x[1:n-1]#Se queda con todas las componenentes menos la primera, equivalente y[-1]-x[-n]
v2
```

```
##   [1]   96 -460  426 -193 -352  -19  459  204  139  -91  735  218  367  795  286
##  [16] -137  205  -95  -73   -8 -316   37 -175  -90 -344    0 -160  209  -17  721
##  [31]  286 -648  441 -677  492   97 -457  -62   13  402  124  -45 -591  436  288
##  [46]   83 -814  688  337 -331 -187  222  222 -590 -165  479 -626   19 -781 -746
##  [61] -126  245  359 -253  -76 -614   24  607 -172 -262  385 -504 -556  462  188
##  [76]  667 -229  124 -594 -240 -580 -611 -171 -461 -884  365  531  546  196 -303
##  [91]  244 -265  633 -400 -828 -548   31  859 -370   48 -446  -65  728  274   14
## [106] -746   54  502  664 -217  164 -286  -15  -87  287  248 -628  550  147 -246
## [121]  -36 -361 -510  -84 -293  212 -345 -547 -632  232  127  236  236 -904   81
## [136]  168   92 -200 -349 -215   56  257  756 -278  334  392 -321 -586 -578 -805
## [151] -164  -81  737  269  119  133   44 -221 -679 -229  308  200  653   24  179
## [166]  -90  100  517   39  519 -273 -233  712  -40 -255   77  341   -4  226  -13
## [181] -215 -151 -103   26 -788 -401   65  610   46  121 -463  549 -450 -721 -463
## [196] -139 -602 -467   -3   55   76  399  443 -588 -444  608 -101  253  588 -314
## [211] -595 -247   89 -534  466  289  642 -455 -537   71  335 -579 -351 -276 -564
## [226]  777  106  176 -331 -218   57  398   60  -59  125 -212 -500 -322  -99    7
## [241] -710    2  601 -175 -368 -587 -858  541  -42
```

```
#voy a ver de donde saldria
head(cbind(y[2:n],x[1:n-1],v2))
```

```
##                v2
## [1,] 235 139   96
## [2,] 316 776 -460
## [3,] 963 537  426
## [4,]  69 262 -193
## [5,]  99 451 -352
## [6,]  70  89  -19
```

otra forma (tidyverse):

```
xyt %>%
  mutate(
    t1=lead(y,n=1),
    t2=x,
    v2=t1-t2
  ) %>%
  pull(v2)
```

```
##   [1]   96 -460  426 -193 -352  -19  459  204  139  -91  735  218  367  795  286
##  [16] -137  205  -95  -73   -8 -316   37 -175  -90 -344    0 -160  209  -17  721
##  [31]  286 -648  441 -677  492   97 -457  -62   13  402  124  -45 -591  436  288
##  [46]   83 -814  688  337 -331 -187  222  222 -590 -165  479 -626   19 -781 -746
##  [61] -126  245  359 -253  -76 -614   24  607 -172 -262  385 -504 -556  462  188
##  [76]  667 -229  124 -594 -240 -580 -611 -171 -461 -884  365  531  546  196 -303
##  [91]  244 -265  633 -400 -828 -548   31  859 -370   48 -446  -65  728  274   14
## [106] -746   54  502  664 -217  164 -286  -15  -87  287  248 -628  550  147 -246
## [121]  -36 -361 -510  -84 -293  212 -345 -547 -632  232  127  236  236 -904   81
## [136]  168   92 -200 -349 -215   56  257  756 -278  334  392 -321 -586 -578 -805
## [151] -164  -81  737  269  119  133   44 -221 -679 -229  308  200  653   24  179
## [166]  -90  100  517   39  519 -273 -233  712  -40 -255   77  341   -4  226  -13
```

```
## [181] -215 -151 -103   26 -788 -401   65  610   46  121 -463  549 -450 -721 -463
## [196] -139 -602 -467   -3   55   76  399  443 -588 -444  608 -101  253  588 -314
## [211] -595 -247   89 -534  466  289  642 -455 -537   71  335 -579 -351 -276 -564
## [226]  777  106  176 -331 -218   57  398   60  -59  125 -212 -500 -322  -99    7
## [241] -710    2  601 -175 -368 -587 -858  541  -42   NA
```

```
#iii) Generar el vector $y_2-y_1,\ldots,y_{250}-y_{249}$.
v3=y[2:n]-y[1:n-1]
v3
```

```
##   [1] -659   81  647 -894   30  -29  661  117 -628  644   88 -300    5  253 -582
##  [16]   13  439 -769  387  423 -756  930 -351  111 -454 -212   16  441  109  169
##  [31] -143 -586  362 -136  609 -693  267 -348   63  261  278 -607 -133  834 -184
##  [46]   83 -581  746  -88 -574 -236  486  -93  -25 -236  787 -787  -61  -49  -18
##  [61]   22  545   77   12 -637  -24  832 -190 -147 -360  729 -835   24  690 -442
##  [76]  389 -295  433 -699  287 -312  -86  775 -749   39  683  127 -147 -123   52
##  [91] -412  -83  725 -475 -411  -10  263  655 -502  102 -530  711   67 -179 -396
## [106]  -47  807 -253  156 -689  560 -728  264    1   86  120 -270  575 -610 -179
## [121]   70  368 -284  697 -343  303 -624   29   65  383  -97 -241  443 -708  564
## [136]   -9   18 -111 -100 -431  387   35  511 -591  225  -52   95 -611  132 -112
## [151]  612 -431  714 -518  -34  177  376 -909   82   90  623 -315  457   32 -626
## [166] -181  500  -29  -29   -9   43 -539  860 -171 -696  310  185  395 -645 -179
## [181]   99  573 -590  471 -657  132  714   91 -692   19  -55  559 -520 -235  128
## [196]  -65 -129  200  655 -425 -360  822  -56 -824  178  616 -798  361  314 -587
## [211]  126   -4  220 -331  537    2  -13 -186 -332  173  193 -397  248  -50   45
## [226]  505 -417  -48   99 -454  220  612 -403   48  417 -498 -327  563 -461  -35
## [241]  -55  609  -51 -517  125  -19 -224  566  301
```

```
cbind(y[2:n],y[1:n-1],y[2:n]-y[1:n-1])
```

```
##        [,1] [,2] [,3]
##   [1,]  235  894 -659
##   [2,]  316  235   81
##   [3,]  963  316  647
##   [4,]   69  963 -894
##   [5,]   99   69   30
##   [6,]   70   99  -29
##   [7,]  731   70  661
##   [8,]  848  731  117
##   [9,]  220  848 -628
##  [10,]  864  220  644
##  [11,]  952  864   88
##  [12,]  652  952 -300
##  [13,]  657  652    5
##  [14,]  910  657  253
##  [15,]  328  910 -582
##  [16,]  341  328   13
##  [17,]  780  341  439
##  [18,]   11  780 -769
##  [19,]  398   11  387
##  [20,]  821  398  423
##  [21,]   65  821 -756
##  [22,]  995   65  930
##  [23,]  644  995 -351
##  [24,]  755  644  111
##  [25,]  301  755 -454
```

```
##  [26,]   89  301 -212
##  [27,]  105   89   16
##  [28,]  546  105  441
##  [29,]  655  546  109
##  [30,]  824  655  169
##  [31,]  681  824 -143
##  [32,]   95  681 -586
##  [33,]  457   95  362
##  [34,]  321  457 -136
##  [35,]  930  321  609
##  [36,]  237  930 -693
##  [37,]  504  237  267
##  [38,]  156  504 -348
##  [39,]  219  156   63
##  [40,]  480  219  261
##  [41,]  758  480  278
##  [42,]  151  758 -607
##  [43,]   18  151 -133
##  [44,]  852   18  834
##  [45,]  668  852 -184
##  [46,]  751  668   83
##  [47,]  170  751 -581
##  [48,]  916  170  746
##  [49,]  828  916  -88
##  [50,]  254  828 -574
##  [51,]   18  254 -236
##  [52,]  504   18  486
##  [53,]  411  504  -93
##  [54,]  386  411  -25
##  [55,]  150  386 -236
##  [56,]  937  150  787
##  [57,]  150  937 -787
##  [58,]   89  150  -61
##  [59,]   40   89  -49
##  [60,]   22   40  -18
##  [61,]   44   22   22
##  [62,]  589   44  545
##  [63,]  666  589   77
##  [64,]  678  666   12
##  [65,]   41  678 -637
##  [66,]   17   41  -24
##  [67,]  849   17  832
##  [68,]  659  849 -190
##  [69,]  512  659 -147
##  [70,]  152  512 -360
##  [71,]  881  152  729
##  [72,]   46  881 -835
##  [73,]   70   46   24
##  [74,]  760   70  690
##  [75,]  318  760 -442
##  [76,]  707  318  389
##  [77,]  412  707 -295
##  [78,]  845  412  433
##  [79,]  146  845 -699
```

```
##  [80,]  433  146  287
##  [81,]  121  433 -312
##  [82,]   35  121  -86
##  [83,]  810   35  775
##  [84,]   61  810 -749
##  [85,]  100   61   39
##  [86,]  783  100  683
##  [87,]  910  783  127
##  [88,]  763  910 -147
##  [89,]  640  763 -123
##  [90,]  692  640   52
##  [91,]  280  692 -412
##  [92,]  197  280  -83
##  [93,]  922  197  725
##  [94,]  447  922 -475
##  [95,]   36  447 -411
##  [96,]   26   36  -10
##  [97,]  289   26  263
##  [98,]  944  289  655
##  [99,]  442  944 -502
## [100,]  544  442  102
## [101,]   14  544 -530
## [102,]  725   14  711
## [103,]  792  725   67
## [104,]  613  792 -179
## [105,]  217  613 -396
## [106,]  170  217  -47
## [107,]  977  170  807
## [108,]  724  977 -253
## [109,]  880  724  156
## [110,]  191  880 -689
## [111,]  751  191  560
## [112,]   23  751 -728
## [113,]  287   23  264
## [114,]  288  287    1
## [115,]  374  288   86
## [116,]  494  374  120
## [117,]  224  494 -270
## [118,]  799  224  575
## [119,]  189  799 -610
## [120,]   10  189 -179
## [121,]   80   10   70
## [122,]  448   80  368
## [123,]  164  448 -284
## [124,]  861  164  697
## [125,]  518  861 -343
## [126,]  821  518  303
## [127,]  197  821 -624
## [128,]  226  197   29
## [129,]  291  226   65
## [130,]  674  291  383
## [131,]  577  674  -97
## [132,]  336  577 -241
## [133,]  779  336  443
```

```
## [134,]   71  779 -708
## [135,]  635   71  564
## [136,]  626  635   -9
## [137,]  644  626   18
## [138,]  533  644 -111
## [139,]  433  533 -100
## [140,]    2  433 -431
## [141,]  389    2  387
## [142,]  424  389   35
## [143,]  935  424  511
## [144,]  344  935 -591
## [145,]  569  344  225
## [146,]  517  569  -52
## [147,]  612  517   95
## [148,]    1  612 -611
## [149,]  133    1  132
## [150,]   21  133 -112
## [151,]  633   21  612
## [152,]  202  633 -431
## [153,]  916  202  714
## [154,]  398  916 -518
## [155,]  364  398  -34
## [156,]  541  364  177
## [157,]  917  541  376
## [158,]    8  917 -909
## [159,]   90    8   82
## [160,]  180   90   90
## [161,]  803  180  623
## [162,]  488  803 -315
## [163,]  945  488  457
## [164,]  977  945   32
## [165,]  351  977 -626
## [166,]  170  351 -181
## [167,]  670  170  500
## [168,]  641  670  -29
## [169,]  612  641  -29
## [170,]  603  612   -9
## [171,]  646  603   43
## [172,]  107  646 -539
## [173,]  967  107  860
## [174,]  796  967 -171
## [175,]  100  796 -696
## [176,]  410  100  310
## [177,]  595  410  185
## [178,]  990  595  395
## [179,]  345  990 -645
## [180,]  166  345 -179
## [181,]  265  166   99
## [182,]  838  265  573
## [183,]  248  838 -590
## [184,]  719  248  471
## [185,]   62  719 -657
## [186,]  194   62  132
## [187,]  908  194  714
```

```
## [188,]  999  908   91
## [189,]  307  999 -692
## [190,]  326  307   19
## [191,]  271  326  -55
## [192,]  830  271  559
## [193,]  310  830 -520
## [194,]   75  310 -235
## [195,]  203   75  128
## [196,]  138  203  -65
## [197,]    9  138 -129
## [198,]  209    9  200
## [199,]  864  209  655
## [200,]  439  864 -425
## [201,]   79  439 -360
## [202,]  901   79  822
## [203,]  845  901  -56
## [204,]   21  845 -824
## [205,]  199   21  178
## [206,]  815  199  616
## [207,]   17  815 -798
## [208,]  378   17  361
## [209,]  692  378  314
## [210,]  105  692 -587
## [211,]  231  105  126
## [212,]  227  231   -4
## [213,]  447  227  220
## [214,]  116  447 -331
## [215,]  653  116  537
## [216,]  655  653    2
## [217,]  642  655  -13
## [218,]  456  642 -186
## [219,]  124  456 -332
## [220,]  297  124  173
## [221,]  490  297  193
## [222,]   93  490 -397
## [223,]  341   93  248
## [224,]  291  341  -50
## [225,]  336  291   45
## [226,]  841  336  505
## [227,]  424  841 -417
## [228,]  376  424  -48
## [229,]  475  376   99
## [230,]   21  475 -454
## [231,]  241   21  220
## [232,]  853  241  612
## [233,]  450  853 -403
## [234,]  498  450   48
## [235,]  915  498  417
## [236,]  417  915 -498
## [237,]   90  417 -327
## [238,]  653   90  563
## [239,]  192  653 -461
## [240,]  157  192  -35
## [241,]  102  157  -55
```

```
## [242,]  711  102  609
## [243,]  660  711  -51
## [244,]  143  660 -517
## [245,]  268  143  125
## [246,]  249  268  -19
## [247,]   25  249 -224
## [248,]  591   25  566
## [249,]  892  591  301
```

Otra forma tidyverse:

```
xyt %>%
  mutate(
    t1=lead(y,n=1),
    t2=y,
    v3=t1-t2
  ) %>%
  pull(v3)
```

```
##   [1] -659   81  647 -894   30  -29  661  117 -628  644   88 -300    5  253 -582
##  [16]   13  439 -769  387  423 -756  930 -351  111 -454 -212   16  441  109  169
##  [31] -143 -586  362 -136  609 -693  267 -348   63  261  278 -607 -133  834 -184
##  [46]   83 -581  746  -88 -574 -236  486  -93  -25 -236  787 -787  -61  -49  -18
##  [61]   22  545   77   12 -637  -24  832 -190 -147 -360  729 -835   24  690 -442
##  [76]  389 -295  433 -699  287 -312  -86  775 -749   39  683  127 -147 -123   52
##  [91] -412  -83  725 -475 -411  -10  263  655 -502  102 -530  711   67 -179 -396
## [106]  -47  807 -253  156 -689  560 -728  264    1   86  120 -270  575 -610 -179
## [121]   70  368 -284  697 -343  303 -624   29   65  383  -97 -241  443 -708  564
## [136]   -9   18 -111 -100 -431  387   35  511 -591  225  -52   95 -611  132 -112
## [151]  612 -431  714 -518  -34  177  376 -909   82   90  623 -315  457   32 -626
## [166] -181  500  -29  -29   -9   43 -539  860 -171 -696  310  185  395 -645 -179
## [181]   99  573 -590  471 -657  132  714   91 -692   19  -55  559 -520 -235  128
## [196]  -65 -129  200  655 -425 -360  822  -56 -824  178  616 -798  361  314 -587
## [211]  126   -4  220 -331  537    2  -13 -186 -332  173  193 -397  248  -50   45
## [226]  505 -417  -48   99 -454  220  612 -403   48  417 -498 -327  563 -461  -35
## [241]  -55  609  -51 -517  125  -19 -224  566  301   NA
```

```
#iv)    Construir el vector $x_1+2x_2-x_3,x_2+2x_3-x_4,...,x_{248}+2x_{249}-x_{250}$.

v4=x[1:248]+2*x[2:249]-x[3:250]
v4
```

```
##   [1] 1154 1588  610 1075  357  -11 1479 -149 1774  955  795  899  478 -279  423
##  [16] 1522  316  219 1748  633 1478 1751 1864 2046  558  282  267 1578  483  150
##  [31] 1865 -223 1574 1734 -243 1844 1191  552 -272 1150  417  998 1061  508  732
##  [46] 2408  949  625 1456  713  580 -316 1826 1148  455 1940   95  944 2187  764
##  [61]  551   27 2052  534  554 2229  245 1006 1016  856  970 1504 1092  518 -431
##  [76]  601 1343 1528 1385 1429 1012 2086 1041 2072 1441  959  369  110 2398  605
##  [91]  671  193 1119 2001 1754 1005 -384 1213 1344  626 1976  579  539 -171 1112
## [106] 2540 1151  246  445 1273  903  538  965  303 -273 1701 1308   77  438 -321
## [121] 1060 1212 1753 1958 1487  920 1165 2177 1357 1242  107  211 1939 1625  918
## [136]  829 1236 2080  883  716  488  -97 1188  967 -448 1404 1396 1183 1566 2137
## [151] 1184  512  192  211  188 1925  562 1358 1092 1111  779  -81 2026 1037  122
## [166] 1276  245 1186 -178 1582 1344   14 1572 1213  767 -153 2123 1053   -3  150
## [181] 2107  998  887 1798 1197 1892 1360  706  -63 1392  536 1005 1686 1851  609
## [196]  823 1096 2026 1632 -112  605  697  977 1688  939  318  264  -86  116 1597
```

```
## [211] 1416  540 1471  658  919 -545 1161 2007  958 -136  807 1489  926 2303  710
## [226]  500  -88 1573 1100  152  704  678  714 1508 1458  834 2249 1407 -221 1065
## [241] 2171  509   59  754 1425 2552   49 1590
```

```
cbind(x[1:248],2*x[2:249],x[3:250],x[1:248]+2*x[2:249]-x[3:250])
```

```
##        [,1] [,2] [,3] [,4]
##   [1,]  139 1552  537 1154
##   [2,]  776 1074  262 1588
##   [3,]  537  524  451  610
##   [4,]  262  902   89 1075
##   [5,]  451  178  272  357
##   [6,]   89  544  644  -11
##   [7,]  272 1288   81 1479
##   [8,]  644  162  955 -149
##   [9,]   81 1910  217 1774
##  [10,]  955  434  434  955
##  [11,]  217  868  290  795
##  [12,]  434  580  115  899
##  [13,]  290  230   42  478
##  [14,]  115   84  478 -279
##  [15,]   42  956  575  423
##  [16,]  478 1150  106 1522
##  [17,]  575  212  471  316
##  [18,]  106  942  829  219
##  [19,]  471 1658  381 1748
##  [20,]  829  762  958  633
##  [21,]  381 1916  819 1478
##  [22,]  958 1638  845 1751
##  [23,]  819 1690  645 1864
##  [24,]  845 1290   89 2046
##  [25,]  645  178  265  558
##  [26,]   89  530  337  282
##  [27,]  265  674  672  267
##  [28,]  337 1344  103 1578
##  [29,]  672  206  395  483
##  [30,]  103  790  743  150
##  [31,]  395 1486   16 1865
##  [32,]  743   32  998 -223
##  [33,]   16 1996  438 1574
##  [34,]  998  876  140 1734
##  [35,]  438  280  961 -243
##  [36,]  140 1922  218 1844
##  [37,]  961  436  206 1191
##  [38,]  218  412   78  552
##  [39,]  206  156  634 -272
##  [40,]   78 1268  196 1150
##  [41,]  634  392  609  417
##  [42,]  196 1218  416  998
##  [43,]  609  832  380 1061
##  [44,]  416  760  668  508
##  [45,]  380 1336  984  732
##  [46,]  668 1968  228 2408
##  [47,]  984  456  491  949
##  [48,]  228  982  585  625
```

```
##  [49,]  491 1170  205 1456
##  [50,]  585  410  282  713
##  [51,]  205  564  189  580
##  [52,]  282  378  976 -316
##  [53,]  189 1952  315 1826
##  [54,]  976  630  458 1148
##  [55,]  315  916  776  455
##  [56,]  458 1552   70 1940
##  [57,]  776  140  821   95
##  [58,]   70 1642  768  944
##  [59,]  821 1536  170 2187
##  [60,]  768  340  344  764
##  [61,]  170  688  307  551
##  [62,]  344  614  931   27
##  [63,]  307 1862  117 2052
##  [64,]  931  234  631  534
##  [65,]  117 1262  825  554
##  [66,]  631 1650   52 2229
##  [67,]  825  104  684  245
##  [68,]   52 1368  414 1006
##  [69,]  684  828  496 1016
##  [70,]  414  992  550  856
##  [71,]  496 1100  626  970
##  [72,]  550 1252  298 1504
##  [73,]  626  596  130 1092
##  [74,]  298  260   40  518
##  [75,]  130   80  641 -431
##  [76,]   40 1282  721  601
##  [77,]  641 1442  740 1343
##  [78,]  721 1480  673 1528
##  [79,]  740 1346  701 1385
##  [80,]  673 1402  646 1429
##  [81,]  701 1292  981 1012
##  [82,]  646 1962  522 2086
##  [83,]  981 1044  984 1041
##  [84,]  522 1968  418 2072
##  [85,]  984  836  379 1441
##  [86,]  418  758  217  959
##  [87,]  379  434  444  369
##  [88,]  217  888  995  110
##  [89,]  444 1990   36 2398
##  [90,]  995   72  462  605
##  [91,]   36  924  289  671
##  [92,]  462  578  847  193
##  [93,]  289 1694  864 1119
##  [94,]  847 1728  574 2001
##  [95,]  864 1148  258 1754
##  [96,]  574  516   85 1005
##  [97,]  258  170  812 -384
##  [98,]   85 1624  496 1213
##  [99,]  812  992  460 1344
## [100,]  496  920  790  626
## [101,]  460 1580   64 1976
## [102,]  790  128  339  579
```

```
## [103,]   64  678  203  539
## [104,]  339  406  916 -171
## [105,]  203 1832  923 1112
## [106,]  916 1846  222 2540
## [107,]  923  444  216 1151
## [108,]  222  432  408  246
## [109,]  216  816  587  445
## [110,]  408 1174  309 1273
## [111,]  587  618  302  903
## [112,]  309  604  375  538
## [113,]  302  750   87  965
## [114,]  375  174  246  303
## [115,]   87  492  852 -273
## [116,]  246 1704  249 1701
## [117,]  852  498   42 1308
## [118,]  249   84  256   77
## [119,]   42  512  116  438
## [120,]  256  232  809 -321
## [121,]  116 1618  674 1060
## [122,]  809 1348  945 1212
## [123,]  674 1890  811 1753
## [124,]  945 1622  609 1958
## [125,]  811 1218  542 1487
## [126,]  609 1084  773  920
## [127,]  542 1546  923 1165
## [128,]  773 1846  442 2177
## [129,]  923  884  450 1357
## [130,]  442  900  100 1242
## [131,]  450  200  543  107
## [132,]  100 1086  975  211
## [133,]  543 1950  554 1939
## [134,]  975 1108  458 1625
## [135,]  554  916  552  918
## [136,]  458 1104  733  829
## [137,]  552 1466  782 1236
## [138,]  733 1564  217 2080
## [139,]  782  434  333  883
## [140,]  217  666  167  716
## [141,]  333  334  179  488
## [142,]  167  358  622  -97
## [143,]  179 1244  235 1188
## [144,]  622  470  125  967
## [145,]  235  250  933 -448
## [146,]  125 1866  587 1404
## [147,]  933 1174  711 1396
## [148,]  587 1422  826 1183
## [149,]  711 1652  797 1566
## [150,]  826 1594  283 2137
## [151,]  797  566  179 1184
## [152,]  283  358  129  512
## [153,]  179  258  245  192
## [154,]  129  490  408  211
## [155,]  245  816  873  188
## [156,]  408 1746  229 1925
```

```
## [157,]  873  458  769  562
## [158,]  229 1538  409 1358
## [159,]  769  818  495 1092
## [160,]  409  990  288 1111
## [161,]  495  576  292  779
## [162,]  288  584  953  -81
## [163,]  292 1906  172 2026
## [164,]  953  344  260 1037
## [165,]  172  520  570  122
## [166,]  260 1140  124 1276
## [167,]  570  248  573  245
## [168,]  124 1146   84 1186
## [169,]  573  168  919 -178
## [170,]   84 1838  340 1582
## [171,]  919  680  255 1344
## [172,]  340  510  836   14
## [173,]  255 1672  355 1572
## [174,]  836  710  333 1213
## [175,]  355  666  254  767
## [176,]  333  508  994 -153
## [177,]  254 1988  119 2123
## [178,]  994  238  179 1053
## [179,]  119  358  480   -3
## [180,]  179  960  989  150
## [181,]  480 1978  351 2107
## [182,]  989  702  693  998
## [183,]  351 1386  850  887
## [184,]  693 1700  595 1798
## [185,]  850 1190  843 1197
## [186,]  595 1686  389 1892
## [187,]  843  778  261 1360
## [188,]  389  522  205  706
## [189,]  261  410  734  -63
## [190,]  205 1468  281 1392
## [191,]  734  562  760  536
## [192,]  281 1520  796 1005
## [193,]  760 1592  666 1686
## [194,]  796 1332  277 1851
## [195,]  666  554  611  609
## [196,]  277 1222  676  823
## [197,]  611 1352  867 1096
## [198,]  676 1734  384 2026
## [199,]  867  768    3 1632
## [200,]  384    6  502 -112
## [201,]    3 1004  402  605
## [202,]  502  804  609  697
## [203,]  402 1218  643  977
## [204,]  609 1286  207 1688
## [205,]  643  414  118  939
## [206,]  207  236  125  318
## [207,]  118  250  104  264
## [208,]  125  208  419  -86
## [209,]  104  838  826  116
## [210,]  419 1652  474 1597
```

```
## [211,]  826  948  358 1416
## [212,]  474  716  650  540
## [213,]  358 1300  187 1471
## [214,]  650  374  366  658
## [215,]  187  732    0  919
## [216,]  366    0  911 -545
## [217,]    0 1822  661 1161
## [218,]  911 1322  226 2007
## [219,]  661  452  155  958
## [220,]  226  310  672 -136
## [221,]  155 1344  692  807
## [222,]  672 1384  567 1489
## [223,]  692 1134  900  926
## [224,]  567 1800   64 2303
## [225,]  900  128  318  710
## [226,]   64  636  200  500
## [227,]  318  400  806  -88
## [228,]  200 1612  239 1573
## [229,]  806  478  184 1100
## [230,]  239  368  455  152
## [231,]  184  910  390  704
## [232,]  455  780  557  678
## [233,]  390 1114  790  714
## [234,]  557 1580  629 1508
## [235,]  790 1258  590 1458
## [236,]  629 1180  975  834
## [237,]  590 1950  291 2249
## [238,]  975  582  150 1407
## [239,]  291  300  812 -221
## [240,]  150 1624  709 1065
## [241,]  812 1418   59 2171
## [242,]  709  118  318  509
## [243,]   59  636  636   59
## [244,]  318 1272  836  754
## [245,]  636 1672  883 1425
## [246,]  836 1766   50 2552
## [247,]  883  100  934   49
## [248,]   50 1868  328 1590
```

#Otra forma :(tidyverse)

```
xyt %>%
  mutate(
    t1=(x),
    t2=lead(x,n=1),
    t3=lead(x,n=2),
    v4=t1+2*t2-t3) %>%
pull(v4)
```

```
##   [1] 1154 1588  610 1075  357  -11 1479 -149 1774  955  795  899  478 -279  423
##  [16] 1522  316  219 1748  633 1478 1751 1864 2046  558  282  267 1578  483  150
##  [31] 1865 -223 1574 1734 -243 1844 1191  552 -272 1150  417  998 1061  508  732
##  [46] 2408  949  625 1456  713  580 -316 1826 1148  455 1940   95  944 2187  764
##  [61]  551   27 2052  534  554 2229  245 1006 1016  856  970 1504 1092  518 -431
##  [76]  601 1343 1528 1385 1429 1012 2086 1041 2072 1441  959  369  110 2398  605
```

```
##  [91]  671  193 1119 2001 1754 1005 -384 1213 1344  626 1976  579  539 -171 1112
## [106] 2540 1151  246  445 1273  903  538  965  303 -273 1701 1308   77  438 -321
## [121] 1060 1212 1753 1958 1487  920 1165 2177 1357 1242  107  211 1939 1625  918
## [136]  829 1236 2080  883  716  488  -97 1188  967 -448 1404 1396 1183 1566 2137
## [151] 1184  512  192  211  188 1925  562 1358 1092 1111  779  -81 2026 1037  122
## [166] 1276  245 1186 -178 1582 1344   14 1572 1213  767 -153 2123 1053   -3  150
## [181] 2107  998  887 1798 1197 1892 1360  706  -63 1392  536 1005 1686 1851  609
## [196]  823 1096 2026 1632 -112  605  697  977 1688  939  318  264  -86  116 1597
## [211] 1416  540 1471  658  919 -545 1161 2007  958 -136  807 1489  926 2303  710
## [226]  500  -88 1573 1100  152  704  678  714 1508 1458  834 2249 1407 -221 1065
## [241] 2171  509   59  754 1425 2552   49 1590   NA   NA
```

```
#v) Calcular la suma de los valores 1/($x_i+y_i$).
sum(1/(x+y))
```

```
## [1] 0.3663756
```

Con el sistema tidyverse:

```
xyt %>%
  mutate(
    t1=x,
    t2=y,
    v4=1/(t1+t2)
  )
```

```
## # A tibble: 250 x 5
##        x     y    t1    t2       v4
##    <int> <int> <int> <int>    <dbl>
##  1   139   894   139   894 0.000968
##  2   776   235   776   235 0.000989
##  3   537   316   537   316 0.00117
##  4   262   963   262   963 0.000816
##  5   451    69   451    69 0.00192
##  6    89    99    89    99 0.00532
##  7   272    70   272    70 0.00292
##  8   644   731   644   731 0.000727
##  9    81   848    81   848 0.00108
## 10   955   220   955   220 0.000851
## # ... with 240 more rows
```

4. Continuando con los vectores x e y anteriores.

```
#i) Determinar las posiciones y valores de los $y_i>600$.
head(cbind(which(y>600),y[which(y>600)])) # nos da las posiciones
```

```
##      [,1] [,2]
## [1,]    1  894
## [2,]    4  963
## [3,]    8  731
## [4,]    9  848
## [5,]   11  864
## [6,]   12  952
```

tidyverse

```
xyt %>%
  mutate(
```

```
    posicion=row_number()
  ) %>%
  filter(y>600) %>%
select(posicion,y) %>%
  head
```

```
## # A tibble: 6 x 2
##   posicion     y
##      <int> <int>
## 1        1   894
## 2        4   963
## 3        8   731
## 4        9   848
## 5       11   864
## 6       12   952
```

```
#ii)    Construir una matriz con las posiciones y valores anteriores, y con los valores de x en esas po
head(cbind(which(y>600),y[which(y>600)],x[which(y>600)]))
```

```
##      [,1] [,2] [,3]
## [1,]    1  894  139
## [2,]    4  963  262
## [3,]    8  731  644
## [4,]    9  848   81
## [5,]   11  864  217
## [6,]   12  952  434
```

```
xyt %>%
  mutate(
    posicion=row_number()
  ) %>%
  filter(y>600) %>%
select(posicion,y,x) %>%
  as.matrix() %>%
  head()
```

```
##      posicion   y   x
## [1,]        1 894 139
## [2,]        4 963 262
## [3,]        8 731 644
## [4,]        9 848  81
## [5,]       11 864 217
## [6,]       12 952 434
```

```
#iii)   Guardar las posiciones como nombres de filas de la matriz anterior.
xyt %>%
  mutate(
    posicion=row_number()
  ) %>%
  filter(y>600) %>%
select(posicion,y,x) %>%
  as.matrix() ->xytmat
rownames(xytmat)=xytmat[,1]
xytmat=xytmat[,-1]
head(xytmat)
```

```
##      y   x
## 1  894 139
## 4  963 262
## 8  731 644
## 9  848  81
## 11 864 217
## 12 952 434
```

```
#iv)    Construir el vector $|x_1-xmedia|^{1/2}$,....,$|x_n-xmedia|^{1/2}$.
xmedia=mean(x)
abs(x-xmedia)^(1/2)
```

```
##   [1] 18.4116268 17.2630241  7.6819268 14.6965302  5.1949976 19.7227787
##   [7] 14.3522820 12.8845644 19.9245577 21.8406044 16.1551230  6.6323450
##  [13] 13.7108716 19.0522440 20.8803257  0.1095445  9.8494670 19.2869904
##  [19]  2.6434826 18.7353142  9.8482486 21.9091762 18.4665102 19.1575573
##  [25] 12.9233123 19.7227787 14.5941084 11.8738368 13.9288190 19.3646069
##  [31]  9.1097750 16.2791892 21.4939061 22.8037716  6.3236066 18.3844500
##  [37] 21.9775340 16.1241434 16.4920587 19.9997000 12.4904764 16.7924983
##  [43] 11.4460474  7.8732458  9.8988888 13.7844840 22.4947105 15.8110088
##  [49]  3.6072150 10.3446605 16.5223485 13.9995714 16.9996471 22.3161825
##  [55] 12.7666754  4.4707941 17.2630241 20.1987128 18.5205831 17.0297387
##  [61] 17.5495869 11.5753186 13.0762380 21.2840786 18.9996842 12.3698019
##  [67] 18.6282581 20.6394767 14.3531181  7.9992500  4.2440547  8.4859885
##  [73] 12.1660182 13.4159606 18.6544365 20.9281628 12.7676153 15.5888422
##  [79] 16.1867847 13.9646697 14.9335863 12.9619443 22.4279290  6.6341541
##  [85] 22.4947105  7.7451921  9.9492713 16.1551230  5.8299228 22.7378979
##  [91] 21.0235106  3.9984997 13.7472906 19.2096851 19.6471881  9.7985713
##  [97] 14.8319924 19.8239249 18.2759952  4.2440547  4.2412262 17.6638614
## [103] 20.3466951 11.7893172 16.5827621 20.9287362 21.0953075 15.9996250
## [109] 16.1860434  8.3658831 10.4408812 12.9995385 13.2660469 10.1483004
## [115] 19.7734165 15.2311523 19.3393899 15.1323495 20.8803257 14.8992617
## [121] 19.0259822 18.1937352 14.0004286 21.6104604 18.2486164 11.4460474
## [127]  8.0007500 17.1759134 21.0953075  5.9989999  5.2903686 19.4419135
## [133]  8.0630019 22.2937659  8.7184861  4.4707941  8.6030227 15.9690952
## [139] 17.4359399 16.1551230 12.0410963 17.6348519 17.2912695 12.0005000
## [145] 15.5880724 18.7879749 21.3310103 10.4408812 15.2647306 18.6550797
## [151] 17.8609070 13.9638104 17.2912695 18.6812205 15.2639444  8.3658831
## [157] 19.8749088 15.7793536 17.0590738  8.3059015  4.1245606 13.7836135
## [163] 13.6377417 21.7947700 17.4925127 14.7644167  9.5922886 18.8145688
## [169]  9.7474099 19.8491310 21.0002857 11.7468294 14.9327827 18.9212050
## [175] 11.0899955 12.0410963 14.9662286 22.7158975 18.9469787 17.2912695
## [181]  1.4184499 22.6055745 11.2688952 14.6632875 19.2876126 10.8172085
## [187] 19.1052872  9.4333451 14.7305126 16.5223485 16.0003750 14.0352414
## [193] 16.7932129 17.8328910 13.7117468 14.1770237 11.5330828 14.0716737
## [199] 19.7233871  9.6947408 21.7942194  4.9002041  8.7171096 11.4460474
## [205] 12.8456997 16.4617132 18.9733497 18.7879749 19.3387694  7.6803646
## [211] 18.6550797  1.9969977 10.9539034 13.1153345 17.0583704 10.5824383
## [217] 21.8629367 20.8089404 13.5281928 15.8741299 17.9718669 13.9288190
## [223] 14.6291490  9.4346171 20.5429307 20.3466951 12.6486363 16.6729721
## [229] 18.1111016 15.4592367 17.1460783  4.7945803  9.3801919  8.8888694
## [235] 17.6638614 12.2886940 10.5835722 22.2937659 13.6743556 18.1104390
## [241] 18.2759952 15.1990789 20.4691964 12.6486363 12.5702824 18.9212050
## [247] 20.1249099 20.6878708 21.3544375 12.2469588
```

```r
#v) Calcular el número de elementos de y que distan menos de 200 del máximo de y.
ymax=max(y)
abs(y-ymax)<200 #(LOS VERDADEROS SON 1 Y LOS FALSOS 0 )
```

```
##   [1]  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE
##  [13] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
##  [49]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE
##  [85] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
##  [97] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
## [109] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [157] FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE
## [181] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE
## [193]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE
## [205] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
## [229] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

```r
sum(abs(y-ymax)<200)
```

```
## [1] 45
```

tidyverse:

```r
xyt %>%
  mutate(
    distancia=abs(y-max(y))) %>%
  filter(distancia<200) %>%
summarise(cuantos=n())
```

```
## # A tibble: 1 x 1
##   cuantos
##     <int>
## 1      45
```

```r
#vi)    ¿Cuántos elementos de x son pares?
length(which(x %% 2==0))#x modulo 2 es verdadero o falso #wich en cuales esto es verdadero
```

```
## [1] 129
```

```r
sum((x %% 2==0))
```

```
## [1] 129
```

tidyverse:

```r
xyt %>%
  filter(x%%2==0) %>%
  count()
```

```
## # A tibble: 1 x 1
```

```
##       n
##   <int>
## 1   129
```

```
#viii)   Seleccionar los elementos de x en posiciones donde y sea múltiplo de 5.
x[(y%%5==0)]
```

```
##  [1] 776 272 955  42 106 958 819 645 337 103  16 140 634 228 458  70 768 298 130
## [20] 740 981 522 418 217 995 462  64 923 408 116 809 458 622 409 495 953 570 124
## [39] 333 254 994 119 179 989 760 796 666 609 118 826   0 672 239 557 629 975 318
## [58]  50
```

tidyverse:

```
xyt %>%
    filter(y%%5==0) %>%
  pull(x)
```

```
##  [1] 776 272 955  42 106 958 819 645 337 103  16 140 634 228 458  70 768 298 130
## [20] 740 981 522 418 217 995 462  64 923 408 116 809 458 622 409 495 953 570 124
## [39] 333 254 994 119 179 989 760 796 666 609 118 826   0 672 239 557 629 975 318
## [58]  50
```

```
#viii)  Ordenar los elementos de x según la ordenación creciente de y.
xyt %>%
  arrange(y) %>% #lo dejo asi o si solo quiero los valores de x
  pull(x)
```

```
##   [1] 711 333 769 676 116 471 790 825 125 416 282 797 643 184 170 302  50 258
##  [19] 981 574 768 631 344 626 984 595 958 451 272 298 554 666 502 809 265 821
##  [37] 409 975 692  16  89 418 333 709 337 826 255 187 646 226 826 611 636 673
##  [55] 458  70 609 496 206 812 945 480 228 923 570 495 256 587 150 843 289 773
##  [73] 207 179 277 867 916  78 955 249 923 358 474 776 961 455 693 883 205 989
##  [91] 836 281 462 375  87  85 442 900 155  89 205 796 537  40 438 734 478 543
## [109]  64 575 567 235 179 260 408 246 806 104 315 167 829 245 254 976 721 590
## [127] 179 200 701 217   3 496 864 650 674 557 661 998 239 634 292 672 852 790
## [145] 218 189 414 933 609 782 873 460 672 125 100 307 934 994 919 587  84 203
## [163] 552 283 458 995 573 911 845 733 340 290 366 291 103   0 115 684 318 931
## [181] 668 124 450 117 743  36 419 641  59 850 216  64 644 984 309 645 196 130
## [199] 444 975 106 379 339 355  42 288 522 118 381 542 395 585 760 351 318 740
## [217] 609  81  52 380 390 811 217 384 408 550 328 139 402 389  42 217 629 491
## [235] 129 229 847 140 622 776 812 953 434 262 836 222 172 119 819 261
```

5. Calcular $1 + (1 + 2) + ... + (1 + 2 + 3.. + 10)$.

```
sum(cumsum(1:10))
```

```
## [1] 220
```

```
cumsum(1:10) %>%
  sum()
```

```
## [1] 220
```

6. Calcular $1 + (2/3) + (2/3)(4/5) + ... + (2/3)(4/5)(6/7) + ... + ((2/3)(4/5)(6/7)...(38/39))$.

```
v6=c(1,seq(from=2,to=38,by=2)/seq(from=3,to=39,by=2))
## no es cumsum como antes poque ahora no se van sumando se van multiplicando.
v6
```

```
##  [1] 1.0000000 0.6666667 0.8000000 0.8571429 0.8888889 0.9090909 0.9230769
##  [8] 0.9333333 0.9411765 0.9473684 0.9523810 0.9565217 0.9600000 0.9629630
## [15] 0.9655172 0.9677419 0.9696970 0.9714286 0.9729730 0.9743590
```

```
cumprod(v6)
```

```
##  [1] 1.0000000 0.6666667 0.5333333 0.4571429 0.4063492 0.3694084 0.3409923
##  [8] 0.3182595 0.2995384 0.2837732 0.2702602 0.2585097 0.2481694 0.2389779
## [15] 0.2307373 0.2232941 0.2165276 0.2103411 0.2046562 0.1994087
```

```
sum(cumprod(v6))
```

```
## [1] 6.976346
```

7. Construir una matriz n x n con 0 en la diagonal, +1 en la mitad triangular superior y -1 en la mitad triangular inferior.

```
n=5
m= diag(0,nrow=n)
m[upper.tri(m)]=1
m[lower.tri(m)]=-1
m
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    1    1    1    1
## [2,]   -1    0    1    1    1
## [3,]   -1   -1    0    1    1
## [4,]   -1   -1   -1    0    1
## [5,]   -1   -1   -1   -1    0
```

8. Construir una matriz con la tabla de multiplicar.

```
i=1:9
j=1:9
A=outer(i,j,"*")
colnames(A)=paste("*",1:9,sep="")
rownames(A)=1:9
A
```

```
##   *1 *2 *3 *4 *5 *6 *7 *8 *9
## 1  1  2  3  4  5  6  7  8  9
## 2  2  4  6  8 10 12 14 16 18
## 3  3  6  9 12 15 18 21 24 27
## 4  4  8 12 16 20 24 28 32 36
## 5  5 10 15 20 25 30 35 40 45
## 6  6 12 18 24 30 36 42 48 54
## 7  7 14 21 28 35 42 49 56 63
## 8  8 16 24 32 40 48 56 64 72
## 9  9 18 27 36 45 54 63 72 81
```

con tidyverse:

```
tibble(f1=1:10) %>%
mutate(tabla1=f1*1,
       tabla2=f1*2,
       tabla3=f1*3,
       tabla4=f1*4,
       tabla5=f1*5,
       tabla6=f1*6,
```

```
       tabla7=f1*7,
       tabla8=f1*8,
       tabla9=f1*9,
       tabla10=f1*10)
```

```
## # A tibble: 10 x 11
##       f1 tabla1 tabla2 tabla3 tabla4 tabla5 tabla6 tabla7 tabla8 tabla9 tabla10
##    <int>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>   <dbl>
##  1     1      1      2      3      4      5      6      7      8      9      10
##  2     2      2      4      6      8     10     12     14     16     18      20
##  3     3      3      6      9     12     15     18     21     24     27      30
##  4     4      4      8     12     16     20     24     28     32     36      40
##  5     5      5     10     15     20     25     30     35     40     45      50
##  6     6      6     12     18     24     30     36     42     48     54      60
##  7     7      7     14     21     28     35     42     49     56     63      70
##  8     8      8     16     24     32     40     48     56     64     72      80
##  9     9      9     18     27     36     45     54     63     72     81      90
## 10    10     10     20     30     40     50     60     70     80     90     100
```

9. Construir una matriz 6x9 con enteros aleatorios en $1, \ldots, 10$.

```
set.seed(12345)
v9=sample((1:10),6*9,replace=T)
M9=matrix(v9,nrow=6,ncol=9)
M9
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## [1,]    3    6    7    3    4    9   10   10    3
## [2,]   10    6    6    9    9    5   10    8    6
## [3,]    8    7    1    4    9    3    3    9    3
## [4,]   10   10    4   10    4    1    3    8    7
## [5,]    8    1    8    7    8    1    3    4   10
## [6,]    2    8   10    2    6    5    1    2    7
```

```
#i) Calcular la suma de cada fila, visualizarlo en una nueva columna.
v1=cbind(M9,apply(M9,1,sum))
v1
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]    3    6    7    3    4    9   10   10    3    55
## [2,]   10    6    6    9    9    5   10    8    6    69
## [3,]    8    7    1    4    9    3    3    9    3    47
## [4,]   10   10    4   10    4    1    3    8    7    57
## [5,]    8    1    8    7    8    1    3    4   10    50
## [6,]    2    8   10    2    6    5    1    2    7    43
```

```
#ii)    Calcular el máximo de cada columna, visualizarlo en una fila nueva.
rbind(M9,apply(M9,2,max))
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## [1,]    3    6    7    3    4    9   10   10    3
## [2,]   10    6    6    9    9    5   10    8    6
## [3,]    8    7    1    4    9    3    3    9    3
## [4,]   10   10    4   10    4    1    3    8    7
## [5,]    8    1    8    7    8    1    3    4   10
## [6,]    2    8   10    2    6    5    1    2    7
## [7,]   10   10   10   10    9    9   10   10   10
```

```r
#iii) Calcular el producto matricial de A por su traspuesta.
M9 %*% t(M9)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]  409  414  277  304  248  250
## [2,]  414  559  380  451  396  293
## [3,]  277  380  319  335  257  201
## [4,]  304  451  335  455  336  257
## [5,]  248  396  257  336  368  252
## [6,]  250  293  201  257  252  287
```