
Лабораторная работа №2

"Элементы множеств. Сложные типы данных и их функции в системе Mathematica"
по курсу "Методы и средства защиты информации"

А-08-19, Балашов С.А.

1. Выбрать имена-идентификаторы для переменных, соответствующих фамилии, имени, отчеству, номеру группы и номеру по списку в группе; в соответствии с принятыми в системе Mathematica правилами:

sssss – имя объекта, заданного пользователем, Sssss – имя объекта, входящего в ядро системы, \$Sssss – имя системного объекта, имена переменных должны начинаться с буквы.

In[1]:=

```
surname  
name  
parentname  
groupnumber  
listnumber
```

Out[1]=

```
surname
```

Out[2]=

```
name
```

Out[3]=

```
parentname
```

Out[4]=

```
groupnumber
```

Out[5]=

```
listnumber
```

2. Присвоить переменным значения соответствующих символьных строк (символьные строки задаются цепочкой символов в кавычках, например "sssss"), используя операцию присваивания: var=value, где var – имя переменной, value – её значение. При таком определении переменные в системе Mathematica являются глобальными.

In[6]:=

```
surname = "Балашов"  
name = "Савва"  
parentname = "Арсеньевич"  
groupnumber = "А-08-19"  
listnumber = "4"
```

Out[6]=

Балашов

Out[7]=

Савва

Out[8]=

Арсеньевич

Out[9]=

А-08-19

Out[10]=

4

3. Создать «пустой» список myID={}, а затем последовательно добавить в него строки с фамилией, именем, отчеством, номером группы и номером по списку в группе, а также число-фамилию и число-имя (из лаб. 1), применив функцию AppendTo[.].

Списки (lists) являются наиболее общим видом сложных (множественных) данных в системе Mathematica. Они представляют совокупность однотипных и разнотипных данных, сгруппированных с помощью фигурных скобок. Например: {1,2,3} – список из трех целых чисел; {a,б,в} – список из трех символьных данных.

В дальнейшем предполагается, что все списки должны иметь оригинальные имена-идентификаторы.

In[27]:=

```
myID = {}
AppendTo[myID, surname]
AppendTo[myID, name]
AppendTo[myID, parentname]
AppendTo[myID, groupnumber]
AppendTo[myID, listnumber]
listf = {1, 0, 11, 0, 24, 14, 2}
listn = {17, 0, 2, 2, 0}
numf = AlgebraicNumber[32, listf]
numn = AlgebraicNumber[32, listn]
AppendTo[myID, numn]
AppendTo[myID, numf]
myID
```

Out[27]=

```
{}
```

Out[28]=

```
{Балашов}
```

Out[29]=

```
{Балашов, Савва}
```

Out[30]=

```
{Балашов, Савва, Арсеньевич}
```

Out[31]=

```
{Балашов, Савва, Арсеньевич, А-08-19}
```

Out[32]=

{Балашов, Савва, Арсеньевич, А-08-19, 4}

Out[33]=

{1, 0, 11, 0, 24, 14, 2}

Out[34]=

{17, 0, 2, 2, 0}

Out[35]=

2 642 422 785

Out[36]=

67 601

Out[37]=

{Балашов, Савва, Арсеньевич, А-08-19, 4, 67 601}

Out[38]=

{Балашов, Савва, Арсеньевич, А-08-19, 4, 67 601, 2 642 422 785}

Out[39]=

{Балашов, Савва, Арсеньевич, А-08-19, 4, 67 601, 2 642 422 785}

4. Выделить тремя способами второй и четвертый элементы из сформированных списков. Для выделения элементов списка list используются двойные квадратные скобки: list[[i]] – выделяет i-ый элемент списка с его начала (если i < 0 – с конца); list[[i,j, ...]] - выделяет i-ый, j-ый и т.д. элементы списка. Функция Part[list,i] – выделяет i-ый элемент списка list.

In[40]:=

```
{myID[[2]], myID[[4]]}
myID[[{2, 4}]]
Part[myID, {2, 4}]
```

Out[40]=

```
{Савва, А-08-19}
```

Out[41]=

```
{Савва, А-08-19}
```

Out[42]=

```
{Савва, А-08-19}
```

5. Проверить, являются ли 5-ый и 7-ой элементы списка myID числом (NumberQ[]) или строкой (StringQ[]).

In[43]:=

```
NumberQ [myID[[5]]]
NumberQ [myID[[7]]]
StringQ [myID[[5]]]
StringQ [myID[[7]]]
```

Out[43]=

```
False
```

Out[44]=

```
True
```

Out[45]=

```
True
```

Out[46]=

```
False
```

6. Определить число элементов списка myID. Использовать функцию Length[list].

In[47]:=

Length[myID]

Out[47]=

7

7. Создать список ранжированных числовых элементов, значения которых лежат в диапазоне от 1 до 100-N. Здесь N – номер по списку в группе, используемая функция Range[imax].

In[48]:=

Range[100 - 4]

Out[48]=

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77,
 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96}
```

8. Сформировать список, состоящий из квадратов целых чисел от 1 до 20. Использовать функцию Table[expr,{i,imax}] – генерирует список значений expr при i, изменяющийся от 1 до imax.

In[49]:=

Table[i * i, {i, 20}]

Out[49]=

```
{1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400}
```

9. Сформировать список, состоящий из степеней числа 2 в диапазоне от 1 до 20.

In[50]:=

Table[2 ^ i, {i, 20}]

Out[50]=

```
{2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096,
 8192, 16384, 32768, 65536, 131072, 262144, 524288, 1048576}
```

10. Создать список целых чисел от 1 до 64 в шестнадцатеричном представлении.

In[51]:=

```
Table[BaseForm[i, 16], {i, 64}]
```

Out[51]=

```
{116, 216, 316, 416, 516, 616, 716, 816, 916, a16, b16, c16, d16, e16, f16, 1016, 1116, 1216, 1316,  
1416, 1516, 1616, 1716, 1816, 1916, 1a16, 1b16, 1c16, 1d16, 1e16, 1f16, 2016, 2116, 2216,  
2316, 2416, 2516, 2616, 2716, 2816, 2916, 2a16, 2b16, 2c16, 2d16, 2e16, 2f16, 3016, 3116,  
3216, 3316, 3416, 3516, 3616, 3716, 3816, 3916, 3a16, 3b16, 3c16, 3d16, 3e16, 3f16, 4016}
```

11. Создать список целых чисел от 1 до 64 в двоичном представлении.

In[52]:=

```
Table[BaseForm[i, 2], {i, 64}]
```

Out[52]=

```
{12, 102, 112, 1002, 1012, 1102, 1112, 10002, 10012, 10102, 10112, 11002, 11012,  
11102, 11112, 100002, 100012, 100102, 100112, 101002, 101012, 101102, 101112,  
110002, 110012, 110102, 110112, 111002, 111012, 111102, 111112, 1000002,  
1000012, 1000102, 1000112, 1001002, 1001012, 1001102, 1001112, 1010002,  
1010012, 1010102, 1010112, 1011002, 1011012, 1011102, 1011112, 1100002,  
1100012, 1100102, 1100112, 1101002, 1101012, 1101102, 1101112, 1110002,  
1110012, 1110102, 1110112, 1111002, 1111012, 1111102, 1111112, 10000002}
```

12. Получить два списка, состоящих из 60-N нулей и 60-N единиц. Использовать функцию Table[expr,{imax}].

In[53]:=

```
Table[0, {i, 60-4}]  
Table[1, {i, 60-4}]
```

Out[53]=

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

Out[54]=

```
{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}
```

13. Подготовить список (rand200) из 200 случайных целых чисел из диапазона 1, 100.

Провести сортировку списка с помощью функции `Sort[list]`, которая располагает элементы списка в каноническом порядке. Определить число элементов со значениями $10+N$, $20+N$, $30+N$. Использовать функцию `Count[list,pattern]` – возвращает количество элементов в списке `list`, которые соответствуют образцу `pattern`.

In[23]:=

```
rand200 = Table[RandomInteger[100], {200}]
Sort[rand200]
Count[rand200, 10 + 4]
Count[rand200, 20 + 4]
Count[rand200, 30 + 4]
```

Out[23]=

```
{79, 8, 42, 82, 33, 2, 23, 18, 42, 42, 63, 88, 5, 13, 29, 1, 49, 59, 6, 35, 73, 0, 98, 82, 55,
84, 55, 50, 87, 2, 15, 1, 74, 66, 62, 59, 92, 35, 0, 5, 50, 24, 74, 99, 65, 73, 56,
76, 83, 75, 6, 26, 21, 75, 41, 69, 89, 56, 23, 36, 23, 37, 60, 85, 2, 63, 51, 69, 17,
22, 93, 2, 1, 96, 25, 27, 63, 85, 51, 42, 98, 12, 43, 28, 88, 92, 66, 49, 49, 3, 20,
28, 5, 31, 53, 88, 79, 79, 33, 4, 25, 64, 10, 33, 86, 84, 41, 51, 43, 86, 83, 17, 60,
75, 33, 6, 53, 20, 61, 9, 91, 58, 87, 8, 27, 52, 80, 42, 66, 49, 53, 68, 97, 43, 82,
99, 94, 25, 7, 62, 75, 62, 25, 6, 84, 5, 39, 31, 52, 80, 30, 2, 37, 5, 39, 4, 60, 80,
59, 60, 86, 28, 94, 23, 58, 28, 54, 15, 80, 50, 98, 81, 81, 90, 36, 17, 78, 10, 40,
40, 11, 20, 98, 77, 81, 86, 25, 87, 91, 35, 75, 23, 40, 73, 40, 55, 84, 79, 64, 21}
```

Out[24]=

```
{0, 0, 1, 1, 1, 2, 2, 2, 2, 2, 3, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 7, 8, 8, 9, 10, 10, 11, 12,
13, 15, 15, 17, 17, 17, 18, 20, 20, 20, 21, 21, 22, 23, 23, 23, 23, 23, 24, 25, 25, 25,
25, 25, 26, 27, 27, 28, 28, 28, 28, 29, 30, 31, 31, 33, 33, 33, 33, 35, 35, 35, 36,
36, 37, 37, 39, 39, 40, 40, 40, 40, 41, 41, 42, 42, 42, 42, 42, 43, 43, 43, 49, 49,
49, 49, 50, 50, 50, 51, 51, 51, 52, 52, 53, 53, 53, 54, 55, 55, 55, 56, 56, 58, 58,
59, 59, 59, 60, 60, 60, 60, 61, 62, 62, 62, 63, 63, 63, 64, 64, 65, 66, 66, 66, 68,
69, 69, 73, 73, 73, 74, 74, 75, 75, 75, 75, 75, 76, 77, 78, 79, 79, 79, 79, 80, 80,
80, 80, 81, 81, 81, 82, 82, 82, 83, 83, 84, 84, 84, 84, 85, 85, 86, 86, 86, 86, 87,
87, 87, 88, 88, 88, 89, 90, 91, 91, 92, 92, 93, 94, 94, 96, 97, 98, 98, 98, 98, 99, 99}
```

Out[25]=

0

Out[26]=

1

Out[27]=

0

14. Получить элементы списка `rand200` с номерами от $10+N$ до $30+N$. Использовать функцию `Take[list,{m,n}]` – возвращает элементы списка с порядковыми номерами от `m` до `n`.

In[12]:=

```
Take[rand200 , {10 + 4, 30 + 4}]
```

Out[12]=

```
{1, 24, 42, 94, 15, 15, 87, 21, 89, 11, 41, 18, 9, 72, 0, 59, 40, 78, 32, 76, 65}
```

15. Провести операцию "поворота" списка влево на 5+N позиций: функция RotateLeft[list,n]- циклический сдвиг списка влево на n позиций.

In[14]:=

```
RotateLeft[rand200 , 5 + 4]
```

Out[14]=

```
{49, 70, 54, 23, 1, 24, 42, 94, 15, 15, 87, 21, 89, 11, 41, 18, 9, 72, 0, 59,
 40, 78, 32, 76, 65, 16, 98, 34, 63, 62, 45, 43, 5, 29, 49, 23, 94, 30, 79, 46,
 62, 98, 8, 24, 0, 87, 61, 86, 16, 16, 72, 49, 25, 35, 35, 23, 91, 3, 52, 70,
 33, 25, 27, 73, 39, 11, 96, 77, 12, 43, 49, 88, 25, 9, 19, 12, 58, 6, 85, 33,
 60, 97, 97, 6, 14, 69, 57, 91, 48, 31, 30, 31, 60, 74, 95, 38, 54, 86, 59, 69,
 9, 86, 12, 62, 84, 37, 95, 85, 68, 41, 50, 14, 60, 52, 68, 33, 54, 8, 67, 31,
 3, 25, 42, 73, 87, 53, 61, 42, 35, 71, 64, 73, 63, 23, 19, 17, 40, 93, 69, 12,
 12, 36, 70, 35, 78, 21, 85, 57, 94, 67, 21, 21, 75, 35, 25, 65, 91, 23, 86,
 21, 97, 36, 99, 23, 90, 10, 26, 99, 18, 39, 88, 99, 15, 36, 97, 88, 46, 27, 93,
 69, 8, 24, 5, 44, 79, 2, 65, 18, 0, 92, 87, 66, 76, 12, 51, 22, 77, 94, 40, 29}
```

16. Провести операцию "поворота" списка вправо на 5+N позиций: функция RotateRight[list,n].

In[15]:=

```
RotateRight[rand200, 5 + 4]
```

Out[15]=

```
{5, 44, 79, 2, 65, 18, 0, 92, 87, 66, 76, 12, 51, 22, 77, 94, 40, 29, 49, 70,
 54, 23, 1, 24, 42, 94, 15, 15, 87, 21, 89, 11, 41, 18, 9, 72, 0, 59, 40, 78,
 32, 76, 65, 16, 98, 34, 63, 62, 45, 43, 5, 29, 49, 23, 94, 30, 79, 46, 62, 98,
 8, 24, 0, 87, 61, 86, 16, 16, 72, 49, 25, 35, 35, 23, 91, 3, 52, 70, 33, 25,
 27, 73, 39, 11, 96, 77, 12, 43, 49, 88, 25, 9, 19, 12, 58, 6, 85, 33, 60, 97,
 97, 6, 14, 69, 57, 91, 48, 31, 30, 31, 60, 74, 95, 38, 54, 86, 59, 69, 9, 86,
 12, 62, 84, 37, 95, 85, 68, 41, 50, 14, 60, 52, 68, 33, 54, 8, 67, 31, 3, 25,
 42, 73, 87, 53, 61, 42, 35, 71, 64, 73, 63, 23, 19, 17, 40, 93, 69, 12, 12, 36,
 70, 35, 78, 21, 85, 57, 94, 67, 21, 21, 75, 35, 25, 65, 91, 23, 86, 21, 97, 36,
 99, 23, 90, 10, 26, 99, 18, 39, 88, 99, 15, 36, 97, 88, 46, 27, 93, 69, 8, 24}
```

17. Определить число копий, для каждого элемента ,содержащегося в списке rand200.

Функция Tally[list].

In[63]:=

Tally[rand200]

Out[63]=

```
{
  {10, 2}, {1, 3}, {8, 4}, {65, 2}, {47, 3}, {51, 1}, {75, 3}, {54, 4}, {28, 7}, {88, 2},
  {50, 2}, {3, 1}, {42, 3}, {25, 2}, {68, 2}, {12, 3}, {60, 2}, {38, 3}, {15, 2},
  {77, 3}, {76, 2}, {73, 2}, {74, 3}, {37, 3}, {17, 4}, {0, 3}, {16, 3}, {93, 5},
  {30, 3}, {85, 3}, {9, 2}, {83, 3}, {21, 5}, {64, 2}, {44, 3}, {86, 2}, {91, 2}, {23, 4},
  {39, 1}, {57, 3}, {92, 2}, {48, 4}, {26, 4}, {100, 2}, {71, 2}, {40, 3}, {70, 1},
  {89, 4}, {67, 2}, {27, 1}, {58, 2}, {14, 3}, {13, 5}, {31, 1}, {35, 3}, {59, 1}, {2, 2},
  {11, 2}, {94, 1}, {32, 1}, {6, 1}, {55, 2}, {53, 1}, {97, 1}, {84, 1}, {78, 3}, {7, 3},
  {49, 3}, {95, 3}, {80, 1}, {34, 2}, {5, 2}, {98, 1}, {20, 1}, {87, 2}, {45, 1}, {63, 2},
  {33, 1}, {18, 1}, {99, 2}, {4, 1}, {96, 1}, {79, 1}, {62, 1}, {66, 1}, {56, 2}, {41, 1}}

```

18. Подготовить три списка (listrng15, listrng20, listrng25), ранжированных в интервалах 1, 15+N; 5, 20+N; 10, 25+N, представляющих собой три конечных множества целых чисел.

In[64]:=

```
listrng15 = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19}
listrng20 =
  {5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24}
listrng25 =
  {10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29}

```

Out[64]=

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19}
```

Out[65]=

```
{5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24}
```

Out[66]=

```
{10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29}
```

19. Во множестве listrng15 найти элементы, которые не содержатся в множествах listrng20, listrng25. Использовать функцию Complement[list,list1,list2,...] – возвращает список list с элементами, которые не содержатся ни в одном из списков list1,list2,...

In[67]:=

```
Complement[listrng15, listrng20, listrng25]
```

Out[67]=

```
{1, 2, 3, 4}
```

20. Определить пересечение множеств listrng15, listrng20, listrng25. Использовать функцию Intersection[list1,list2,...] – возвращает упорядоченный список элементов, общих для всех списков listi.

In[68]:=

```
Intersection[listrng15, listrng20, listrng25]
```

Out[68]=

```
{10, 11, 12, 13, 14, 15, 16, 17, 18, 19}
```

21. Провести операцию конкатенации для множеств listrng15, listrng20, listrng25. Использовать функцию Join[list1,list2,...] – объединяет множества (списки) в единую цепочку.

In[69]:=

```
Join[listrng15, listrng20, listrng25]
```

Out[69]=

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 5, 6,
 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 10,
 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29}
```

22. Провести операцию объединения множеств listrng15, listrng20, listrng25. Использовать функцию Union [list1,list2,...] – удаляет повторяющиеся элементы списков и возвращает отсортированный список всех различающихся между собой элементов, принадлежащих любому из данных списков listi.

In[70]:=

```
Union[listrng15, listrng20, listrng25]
```

Out[70]=

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,
 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29}
```

23. Удалить повторяющиеся элементы из списка rand200. Использовать функцию Union [list] – возвращает отсортированный вариант списка list, в котором опущены все повторяющиеся элементы

In[32]:=

```
Union[rand200]
Length[rand200] – Length[Union[rand200]]
```

Out[32]=

```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 17, 18, 20, 21, 22, 23, 24, 25,
 26, 27, 28, 29, 30, 31, 33, 35, 36, 37, 39, 40, 41, 42, 43, 49, 50, 51, 52, 53,
 54, 55, 56, 58, 59, 60, 61, 62, 63, 64, 65, 66, 68, 69, 73, 74, 75, 76, 77, 78,
 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 96, 97, 98, 99}
```

Out[33]=

117