

```

1: program Statistics_Module_4;
2: //*****
3: //all 4 examples discuss a combination lock (Zahlenschloss)
4: //loc's = 199, ex. 212_ , adapt from _92
5: //...we discuss the 4 types of stat with a combination lock (zahlenschloss):
6:
7: {1. Permutation = n!
8: 2. Permutation(Variation without repeating) = nPr = n!/(n-k)!
9: 3. Combination (binominal coefficient)= nCr = nPr / k!
10: 4. Variation(repeating) = n^k
11:
12: where,
13:     n,k are non negative integers and k<=n.
14:     k is the size of each permutation.
15:     n is the size of a set from which elements are permuted.
16:     ! is the factorial operator.
17:
18: 1 Anordnungen (Permutationen)
19: 2 Auswählen mit Beachtung der Reihenfolge (Variationen)
20:     * 2.1 Variation ohne Zurücklegen
21:     * 4.1 Variation mit Zurücklegen
22: 3 Auswählen ohne Beachtung der Reihenfolge (Kombinationen)
23:
24: -----
25: {1. ex. lock of 4 numbers [0..3] permutatation, ex. [0123], 4! = 24 results
26: 2. ex. 4 lock with 10 numbers [0..9] permutation, ex. [3579],
27:                                     nPr(10,4) = 5040 results
28: 3. ex. 21 lock with 4 possibilities, ex. [000010000100010100000]
29:                                     nCr(20,4) = 4845 results
30: 4. ex. 4 lock with 10 numbers (norm case), ex. [1122], 10^4 = 10000 results
31: -----}
32:
33: const
34:     DIM = 660;
35:     PI2 = 6.28318530717958647692528676655901;
36:     E = 2.718281828459;
37:     SEP = '-----';
38:     BODY = 23;
39:     EMOTION = 28;
40:     SPIRIT = 33;
41:
42: type TBIOVector = array[0..DIM] of TPoint;
43:     Tperm = array[1..4] of byte;
44:     TCube = array[1..6] of byte;
45:
46: var
47:     hnd: THandle;
48:     mname, mdes: string;
49:     mfile1: TStringList;
50:     st_string: shortstring;
51:     time1, time2, diff: TDateTime;
52:
53: // 1. Permutation n!
54: //-----
55: Procedure One_Permutation_4;
56: var
57:     i,k,l,m: byte;
58:     count: integer;
59: begin
60:     count:= 0;
61:     for i:= 0 to 3 do
62:         for k:= 0 to 3 do
63:             for l:= 0 to 3 do
64:                 for m:= 0 to 3 do begin
65:                     if (i<>k) and (i<>l) and (i<>m) and (k<>l) and (k<>m) and (l<>m)
66:                         then begin
67:                             inc(count);
68:                             Writeln(Format('Case: %d - %d%d%d%d',[count,i,k,l,m]))
69:                             //inc(i); inc(k);
70:                         end;
71:                     end;
72:                 Writeln(IntToStr(count)+' Permutations')
73:                 Writeln('All Permutations of 0..3: '+intToStr(round(Fact(4))))
74:                 Writeln(SEP)
75:             end;
76:

```

```

77:
78: // 2. Permutation(Variation without repeating) = nPr = n!/(n-k)!
79: //-----
80: Procedure Two_Permutation_4_of_10;
81: var
82:   i,k,l,m: byte;
83:   count: integer;
84: begin
85:   count:= 0;
86:   for i:= 0 to 9 do
87:     for k:= 0 to 9 do
88:       for l:= 0 to 9 do
89:         for m:= 0 to 9 do begin
90:           if (i<>k) and (i<>l) and (i<>m) and (k<>l) and (k<>m) and (l<>m)
91:             then begin
92:               inc(count);
93:               Writeln(Format('Case: %d - %d%d%d%d', [count,i,k,l,m]));
94:               //inc(i); inc(k);
95:             end;
96:           end;
97:           Writeln(intToStr(count)+' Permutations')
98:           Writeln('All Permutations 4 of 10: '+intToStr(round(Fact(10)/Fact(10-4))))
99:           Writeln(SEP)
100: end;
101:
102:
103: // 3. Combination (binominal coefficient)= nCr = nPr / k!
104: //-----
105: procedure Three_Combination_without_Order_4_of_20;
106: //binominal coefficient)= nCr = nPr / k!
107: //4 of 20 = 4845 = NCR(20,4) on a calculator or a lotto 4 of 20
108: var
109:   i,k,l,m,n,o,p,q,r,s,t,u,v,w,ql,r1,s1,t1,ul,v1: byte;
110:   count: integer;
111: begin
112:   i:=1; k:=1; l:=1; m:=1; n:=1; o:= 1; p:=1;
113:   q:=1; r:=1; s:=1; t:=1; u:=1; v:= 1; w:=1;
114:   ql:=1; r1:=1; s1:=1; t1:=1; ul:=1; v1:= 1;
115:   count:= 0;
116:   for i:= 0 to 1 do
117:     for k:= 0 to 1 do
118:       for l:= 0 to 1 do
119:         for m:= 0 to 1 do
120:           for n:= 0 to 1 do
121:             for o:= 0 to 1 do
122:               for p:= 0 to 1 do
123:                 for q:= 0 to 1 do
124:                   for r:= 0 to 1 do
125:                     for s:= 0 to 1 do
126:                       for t:= 0 to 1 do
127:                         for u:= 0 to 1 do
128:                           for v:= 0 to 1 do
129:                             for w:= 0 to 1 do
130:                               for ql:= 0 to 1 do
131:                                 for r1:= 0 to 1 do
132:                                   for s1:= 0 to 1 do
133:                                     for t1:= 0 to 1 do
134:                                       for ul:= 0 to 1 do
135:                                         for v1:= 0 to 1 do
136:                                           if i+k+l+m+n+o+p+q+r+s+t+u+v+w+ql+r1+s1+t1+ul+v1 = 4 then begin
137:                                             inc(count);
138:                                             Writeln(Format('Case: %d - %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d',
139: [count,i,k,l,m,n,o,p,q,r,s,t,u,v,w,ql,r1,s1,t1,ul,v1]));
140:                                           end;
141:                                           Writeln(inttostr(count)+' All combinations in a Set')
142:                                           Writeln(SEP)
143:                                         end;
144:
145:
146: // 4. Variation (repeating) = n^k
147: //-----
148: procedure Four_Variation_Lock_4;
149: //solution is n^k = 10^4 = 10000
150: var
151:   i,k,l,m: byte;
152:   count: integer;

```

```

153: begin
154:   i:=1; k:=1; l:=1; m:=1;
155:   count:= 0;
156:   for i:= 0 to 9 do
157:     for k:= 0 to 9 do
158:       for l:= 0 to 9 do
159:         for m:= 0 to 9 do begin
160:           inc(count);
161:           writeln(Format('Case: %d - %d%d%d%d',[count, i,k,l,m]))
162:         end;
163:       Writeln(inttostr(count)+' All Variations of a Lock')
164:       Writeln(SEP)
165:     end;
166:   end;
167:
168: // main calc & plot
169: begin
170:   //Sudoku_permutation_1;
171:   maxForm1.ShellStyle1Click(self);
172:   st_String:= 'STATISTIC MODULE 4';
173:   Writeln(st_String)
174:   time1:= time;
175:
176:   //Showmessage(' First a 4 of 4 permutation')
177:   One_Permutation_4;
178:   //Showmessage(' Second a 4 of 10 permutation')
179:   Two_Permutation_4_of_10;
180:   //Showmessage(' Third a 20 of 4 combination')
181:   Three_Combination_without_Order_4_of_20;
182:   //Showmessage(' Fourth a 4 of 10 variation')
183:   Four_Variation_Lock_4;
184:
185:   time2:= time;
186:   Diff:= time2 - time1;
187:   writeln(FormatDateTime(' " Statistic Module run time is:" nn:ss:zzz',Diff));
188:   Writeln(st_String)
189:   //maxForm1.SaveOutput1Click(self);
190:   writeln('maXbox V: '+GetVersionString('maxbox3.exe'))
191:   //hnd:= ExecuteFile('maxbox3.exe',' ',ExePath,1)
192:   //writeln(inttostr(hnd))
193:   getAssociatedProgram('.pdf',mname, mdes)
194:   writeln(mname + mdes)
195:   mfile1:= TStringlist.create;
196:   //FilesFromWildcard(Exepath, '21*.txt',mfile1, true, true, true)
197:   //Writeln(mfile1.text)
198:   mfile1.Free;
199: end.
200:
201: just maXbox
202:
203:   _____
204:   | | | | | | | | | |
205:   | | | | | | | | | |
206:   | | | | | | | | | |
207:   | | | | | | | | | |
208:   | | | | | | | | | |
209:   | | | | | | | | | |

```