

MLC Lab Visit - Combinatorics in Maple

Mth 355 Fall 2001 Assignment 3. Due Oct 17.

Mth 355 (a.k.a. Mth 399) Oct 10 2001 Maple 6

Bent E. Petersen

Filename: 355f2001-assign-003-combinatorics.mws

This worksheet contains a few comments on some of Maple's combinatorial functions. There are also a few problems for you. The problems constitute Assignment 3.

This worksheet was composed fairly quickly and is unlikely to be free of errors. If you find a serious error be sure to mention it in your solution report.

```
> restart;
```

```
> with(combinat):
```

```
Warning, the protected name Chi has been redefined and unprotected
```

Stirling Numbers of the second kind, $S(r,n)$

$S(r,n)$, denoted by `stirling2(r,n)` in Maple, is the number of partitions of a set of cardinality r into n (nonempty) subsets. Clearly we must have $n \leq r$. You can think of it as the number of ways to place r distinguishable objects into n indistinguishable buckets with no bucket empty. By convention there is one partition of the empty set (the empty partition) with no subsets.

```
> S(0,0) = stirling2(0,0); S(4,0) = stirling2(4,0); S(4,2) =  
stirling2(4,2); S(7,3) = stirling2(7,3); S(21,4) =  
stirling2(21,6);
```

$$S(0, 0) = 1$$

$$S(4, 0) = 0$$

$$S(4, 2) = 7$$

$$S(7, 3) = 301$$

$$S(21, 4) = 26585679462804$$

The last two entries above show that a direct by-hand count may be difficult or infeasible.

Bell Numbers

The Maple function call `bell(n)` returns the n -th Bell number. The n -th Bell number is the number of ways that a set with n elements can be partitioned into a union of disjoint (nonempty) subsets. For example, the set $\{1,2,3,4\}$ can be partitioned in the following ways:

$\{\{1\},\{2\},\{3\},\{4\}\},$
 $\{\{1,2\},\{3\},\{4\}\}, \{\{2,3\},\{1\},\{4\}\}, \{\{3,4\},\{1\},\{2\}\}, \{\{1,4\},\{2\},\{3\}\}, \{\{1,3\},\{2\},\{4\}\}$
 $\{\{1,2\},\{3,4\}\}, \{\{2,3\},\{4,1\}\}, \{\{3,4\},\{1,2\}\}, \{\{1,4\},\{2,3\}\},$
 $\{\{1,2,3\},\{4\}\}, \{\{2,3,4\},\{1\}\}, \{\{3,4,1\},\{2\}\}, \{\{4,2,1\},\{3\}\},$
 $\{\{1,2,3,4\}\},$

so in 15 ways. Let's see what Maple says

```
> bell(4);
```

15

The Bell numbers grow fairly quickly with n ,

```
> 'bell(8)' = bell(8); 'bell(10)' = bell(10); 'bell(20)' = bell(20);
```

bell(8) = 4140
bell(10) = 115975
bell(20) = 51724158235372

Clearly a direct enumeration like the one we did above rapidly becomes impractical. From the definition it is clear that

```
> 'bell(r)' = Sum('S(r,n)', n=0..r);
```

$$\text{bell}(r) = \sum_{n=0}^r S(r, n)$$

Let's check this relation for the case enumerated above:

```
> sum(stirling2(4,n), n=0..4);
```

15

There are a number of interpretations of Bell numbers. For example, consider the problem of putting r distinguishable objects into r indistinguishable buckets (in no particular order, since otherwise we could distinguish them). Each arrangement may be considered a partition of the set of objects. Thus the number of ways is $\text{bell}(n)$.

Number of factorizations of a square-free natural number

A natural number is square-free if it has no square factors. Such a number is a product of distinct primes. In this case we can find all factorization by partitioning the set of prime factors and multiplying out the ones corresponding to subsets in the partition. Thus the number of factorizations is $\text{bell}(r)$ where r is the number of (distinct) prime factors.

For example, 210 is the product of the primes 2, 3, 5 and 7 and so has $\text{bell}(4) = 15$ factorizations.

Problem 1. (by hand) List all the factorizations of 210.

Construction of the set of partitions

Sometimes we want an explicit list of all partitions of a set so we can perform some operation for each partition by stepping through the list. Here is a procedure which returns the set of all partitions of the set $\{1,2,3,\dots,n\}$. We can easily convert to a list instead (see below).

```
> parts:=proc(n)
>   local P,A,B,C; option remember;
>   if n=1 then
>     P:={{1}}
>   elif n=2 then
>     P:={ {{1},{2}}, {{1,2}} };
>   elif n>2 then
>     P:={};
>     for A in parts(n-1) do
>       P:= P union { A union {{n}} };
>       for B in A do
>         C:= ( A minus {B} ) union { B union {n} };
>         P:= P union { C };
>       od;
>     od;
>   else
>     P:= FAIL;
>   fi;
>   return P;
> end;
```

Here is an example

```
> parts(4);
{{{2}, {3}, {4}, {1}}, {{3}, {2, 4}, {1}}, {{2}, {3, 4}, {1}}, {{2}, {3}, {1, 4}},
  {{2, 3}, {4}, {1}}, {{2, 3, 4}, {1}}, {{2, 3}, {1, 4}}, {{2}, {1, 3}, {4}},
  {{1, 3}, {2, 4}}, {{2}, {1, 3, 4}}, {{1, 2}, {3}, {4}}, {{3}, {1, 2, 4}}, {{1, 2}, {3, 4}},
  {{1, 2, 3}, {4}}, {{1, 2, 3, 4}}}
```

By using the `subs()` command we can find the partitions required to solve problem 1. We convert

parts(4) to a list first so Maple will not alter the order the partitions appear in. That makes it easier to check our work by visual inspection.

```
>  
> subs(4=7,3=5,2=3,1=2,[op(parts(4))]);  
[{{2},{3},{7},{5}},{{2},{3,7},{5}},{{2},{3},{5,7}},{{3},{2,7},{5}},  
  {{2},{7},{3,5}},{{2},{3,5,7}},{{2,7},{3,5}},{{3},{7},{2,5}},  
  {{3,7},{2,5}},{{3},{2,5,7}},{{2,3},{7},{5}},{{2,3,7},{5}},{{2,3},{5,7}},  
  {{7},{2,3,5}},{{2,3,5,7}}]
```

We can index over the set of partitions, for example

```
> for A in parts(3) do A; od;  
      {{2},{3},{1}}  
      {{2,3},{1}}  
      {{2},{1,3}}  
      {{1,2},{3}}  
      {{1,2,3}}
```

If a list is preferred we can easily convert as above. For example

```
> L:= [op(parts(3))];  
      L:= [{{2},{3},{1}},{{2,3},{1}},{{2},{1,3}},{{1,2},{3}},{{1,2,3}}]
```

Now we index over the partitions as for any array. For example

```
> for k from 1 to bell(3) do L[k]; od;  
      {{2},{3},{1}}  
      {{2,3},{1}}  
      {{2},{1,3}}  
      {{1,2},{3}}  
      {{1,2,3}}
```

The number of partitions grows rapidly, so be careful when using parts(n)

```
> parts(5);  
[{{2},{3},{4},{5},{1}},{{3},{4},{2,5},{1}},{{2},{4},{3,5},{1}},  
  {{2},{3},{4,5},{1}},{{2},{3},{4},{1,5}},{{3},{2,4},{5},{1}},  
  {{2,4},{3,5},{1}},{{3},{2,4,5},{1}},{{3},{2,4},{1,5}},  
  {{2},{3,4},{5},{1}},{{3,4},{2,5},{1}},{{2},{3,4,5},{1}},
```

```

{{2}, {3, 4}, {1, 5}}, {{2}, {3}, {1, 4}, {5}}, {{3}, {1, 4}, {2, 5}},
{{2}, {1, 4}, {3, 5}}, {{2}, {3}, {1, 4, 5}}, {{2, 3}, {4}, {5}, {1}},
{{4}, {2, 3, 5}, {1}}, {{2, 3}, {4, 5}, {1}}, {{2, 3}, {4}, {1, 5}}, {{2, 3, 4}, {5}, {1}},
{{2, 3, 4, 5}, {1}}, {{2, 3, 4}, {1, 5}}, {{2, 3}, {1, 4}, {5}}, {{1, 4}, {2, 3, 5}},
{{2, 3}, {1, 4, 5}}, {{2}, {1, 3}, {4}, {5}}, {{1, 3}, {4}, {2, 5}}, {{2}, {4}, {1, 3, 5}},
{{2}, {1, 3}, {4, 5}}, {{1, 3}, {2, 4}, {5}}, {{2, 4}, {1, 3, 5}}, {{1, 3}, {2, 4, 5}},
{{2}, {1, 3, 4}, {5}}, {{1, 3, 4}, {2, 5}}, {{2}, {1, 3, 4, 5}}, {{1, 2}, {3}, {4}, {5}},
{{3}, {4}, {1, 2, 5}}, {{1, 2}, {4}, {3, 5}}, {{1, 2}, {3}, {4, 5}}, {{3}, {1, 2, 4}, {5}},
{{1, 2, 4}, {3, 5}}, {{3}, {1, 2, 4, 5}}, {{1, 2}, {3, 4}, {5}}, {{3, 4}, {1, 2, 5}},
{{1, 2}, {3, 4, 5}}, {{1, 2, 3}, {4}, {5}}, {{4}, {1, 2, 3, 5}}, {{1, 2, 3}, {4, 5}},
{{1, 2, 3, 4}, {5}}, {{1, 2, 3, 4, 5}}

```

Power Set

In a previous worksheet we wrote a procedure to compute the power set of a set. A much easier way is to use Maple's procedure `choose()`:

```

> powset := A -> choose (A) ;
                                     powset := choose
> powset ({a, b, c, d}) ;
{{d, a, b, c}, {a, b, c}, {b, c}, {d, b, c}, {c}, {d, c}, {a, c}, {d, a, c}, {d}, {a}, {d, a},
 {b}, {d, b}, {a, b}, {d, a, b}, { }}

```

Another way is just to use Maple's builtin function `powerset()` (but that is less fun).

```

> powerset ({a, b, c, d}) ;
{{d, a, b, c}, {a, b, c}, {b, c}, {d, b, c}, {c}, {d, c}, {a, c}, {d, a, c}, {d}, {a}, {d, a},
 {b}, {d, b}, {a, b}, {d, a, b}, { }}

```

However `choose()` has more capabilities. Thus we can find all the subsets of a given cardinality

```

> choose ({a, b, c, d, e}, 3) ;
{{a, b, c}, {d, b, c}, {d, a, c}, {d, a, b}, {d, e, a}, {d, e, b}, {d, e, c}, {e, a, b}, {e, a, c},
 {e, b, c}}

```

The procedure `choose()` also works on lists. For example

```

> choose ([a, a, b, b, b], 3) ;
[[a, a, b], [a, b, b], [b, b, b]]

```

```
> choose([a,a,b,b,b,c],3);
[[a,b,c],[a,a,b],[a,b,b],[b,b,b],[a,a,c],[b,b,c]]
```

We can also compute the list of all sublists

```
> choose([a,a,b,b,b,b]);
[[],[a],[b],[a,b],[b,b],[a,b,b],[b,b,b],[a,b,b,b],[b,b,b,b],[a,b,b,b,b],[a,a],
[a,a,b],[a,a,b,b],[a,a,b,b,b],[a,a,b,b,b,b]]
> choose([a,a,b,b,b,c]);
[[],[a],[b],[a,b],[b,b],[a,b,b],[c],[a,c],[b,c],[a,b,c],[b,b,c],[a,b,b,c],
[b,b,b],[a,b,b,b],[b,b,b,c],[a,b,b,b,c],[a,a],[a,a,b],[a,a,b,b],[a,a,c],
[a,a,b,c],[a,a,b,b,c],[a,a,b,b,b],[a,a,b,b,b,c]]
```

The procedure powerset() also works on lists. Thus

```
> powerset([a,a,b,b,b,b]);
[[],[a],[b],[a,b],[b,b],[a,b,b],[b,b,b],[a,b,b,b],[b,b,b,b],[a,b,b,b,b],[a,a],
[a,a,b],[a,a,b,b],[a,a,b,b,b],[a,a,b,b,b,b]]
> powerset([a,a,b,b,b,c]);
[[],[a],[b],[a,b],[b,b],[a,b,b],[c],[a,c],[b,c],[a,b,c],[b,b,c],[a,b,b,c],
[b,b,b],[a,b,b,b],[b,b,b,c],[a,b,b,b,c],[a,a],[a,a,b],[a,a,b,b],[a,a,c],
[a,a,b,c],[a,a,b,b,c],[a,a,b,b,b],[a,a,b,b,b,c]]
```

It seems there are always many ways to do any one thing in Maple!

Note maple provides a procedure subsets() which allows one to iterate over the power set without actually computing and storing the whole power set. In large calculations you will need to know about subsets() (and other iterators).

Epimorphisms

Let X be a set with cardinality r and let Y be a set with cardinality n . Suppose $n \leq r$ and let $E(r,n)$ be the number of epimorphisms $f: X \rightarrow Y$. Note each epimorphism gives rise to a partition of X into n subsets indexed by the elements of Y . Composing an epimorphism with a permutation of Y gives rise to another epimorphism which corresponds to the same partition. Thus we see that $Epi(r,n) = S(r,n) * n!$

```
> Epi := (r,n) ->stirling2(r,n)*n!;
Epi := (r, n) → stirling2(r, n) n!
```

Thus the number of epimorphisms of $\{1,2,3,4,5,6\}$ onto $\{1,2,3\}$ is

```
> Epi(6,3);
```

540

Compositions and Partitions of a Positive Integer

A k -composition of a positive integer n is a k -tuple of positive integers summing to n . The order matters. The Maple function call `composition(n,k)` returns a list of all the k -compositions of n . For example

```
> composition(5,3);
```

```
[[2, 2, 1], [3, 1, 1], [1, 3, 1], [2, 1, 2], [1, 2, 2], [1, 1, 3]]
```

The number of such compositions is returned by `numbcomp(n,k)`

```
> numbcomp(5,3);
```

6

We can think of `numbcomp(n,k)` as the number of ways to put n ones in k distinguishable buckets with the proviso that each bucket must contain at least one one. Think about it.

Problem 2. Use Maple to verify $\text{numbcomp}(n,k) = \text{binomial}(n-1,k-1)$ for a few special cases. Then prove it (by hand) for the general case..

Maple uses `binomial(n,k)` for the binomial numbers.

A k -partition of a positive integer n is a set of k positive integers summing to n . The order does not matter. The Maple function call `partition(n)` returns a list of all the partitions of n (each given as a list, rather than a set). For example

```
> partition(5);
```

```
[[1, 1, 1, 1, 1], [1, 1, 1, 2], [1, 2, 2], [1, 1, 3], [2, 3], [1, 4], [5]]
```

The number of partitions is given by `numbpart(n)`

```
> numbpart(5);
```

7

We can get a list of all compositions without regard to length by concatenating lists

```
> allcomp:=proc(n)
```

```

> local k,L;
> L:=[];
> for k from 1 to n do
>   L:= [ op(L),op(composition(n,k)) ];
> od;
> return L;
> end:

```

Thus the list of all compositions of 5 is given by

```

> allcomp(5); nops(%);
[[5], [3, 2], [4, 1], [2, 3], [1, 4], [2, 2, 1], [3, 1, 1], [1, 3, 1], [2, 1, 2], [1, 2, 2], [1, 1, 3],
 [1, 1, 1, 2], [2, 1, 1, 1], [1, 2, 1, 1], [1, 1, 2, 1], [1, 1, 1, 1, 1]]
16

```

The number of compositions of n of any length is given by

```

> k:='k': sum(binomial(n-1,k-1),k=1..n);
2(n-1)

```

Maple returns the list of partitions of n in a unique order. Two procedures encodepart() and decodepart() are available to access the list:

```

> L:=partition(5);
L := [[1, 1, 1, 1, 1], [1, 1, 1, 2], [1, 2, 2], [1, 1, 3], [2, 3], [1, 4], [5]]
> 'decodepart(5,3)' = decodepart(5,3), 'L[3]'=L[3];
decodepart(5,3) = [1, 2, 2], L3 = [1, 2, 2]
> 'encodepart([1,2,2])'=encodepart([1,2,2]);
encodepart([1,2,2]) = 3

```

In addition there are procedures firstpart(), nextpart(), lastpart(), prevpart() and conjpart() which refer to the list of partitions in its canonical order. We can even ask for a random partition

```

> randpart(20);
[1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 6]

```

r-permutations

The procedure permute(m,r) returns a list of all r-permutations of {1,2,...,m}. The procedure numbperm(m,r) returns the number of r-permutations. Thus

```
> permute(5,3); numbperm(5,3);
```

```
[[1, 2, 3], [1, 2, 4], [1, 2, 5], [1, 3, 2], [1, 3, 4], [1, 3, 5], [1, 4, 2], [1, 4, 3], [1, 4, 5],  
 [1, 5, 2], [1, 5, 3], [1, 5, 4], [2, 1, 3], [2, 1, 4], [2, 1, 5], [2, 3, 1], [2, 3, 4], [2, 3, 5], [2, 4, 1],  
 [2, 4, 3], [2, 4, 5], [2, 5, 1], [2, 5, 3], [2, 5, 4], [3, 1, 2], [3, 1, 4], [3, 1, 5], [3, 2, 1], [3, 2, 4],  
 [3, 2, 5], [3, 4, 1], [3, 4, 2], [3, 4, 5], [3, 5, 1], [3, 5, 2], [3, 5, 4], [4, 1, 2], [4, 1, 3], [4, 1, 5],  
 [4, 2, 1], [4, 2, 3], [4, 2, 5], [4, 3, 1], [4, 3, 2], [4, 3, 5], [4, 5, 1], [4, 5, 2], [4, 5, 3], [5, 1, 2],  
 [5, 1, 3], [5, 1, 4], [5, 2, 1], [5, 2, 3], [5, 2, 4], [5, 3, 1], [5, 3, 2], [5, 3, 4], [5, 4, 1], [5, 4, 2],  
 [5, 4, 3]]
```

60

Like most Maple functions `permute()` is polymorphic. If we pass a list `L`, say `permute(L,r)`, we get all the `r`-permutations of the elements of the list `L`

```
> permute([a,b,c,d],3); numbperm([a,b,c,d],3);
```

```
[[a, b, c], [a, b, d], [a, c, b], [a, c, d], [a, d, b], [a, d, c], [b, a, c], [b, a, d], [b, c, a], [b, c, d],  
 [b, d, a], [b, d, c], [c, a, b], [c, a, d], [c, b, a], [c, b, d], [c, d, a], [c, d, b], [d, a, b], [d, a, c],  
 [d, b, a], [d, b, c], [d, c, a], [d, c, b]]
```

24

Note `permute()` and `numbperm()` even work correctly on lists with repeated elements

```
> permute([a,b,c,c],3); numbperm([a,b,c,c],3);
```

```
[[a, b, c], [a, c, b], [a, c, c], [b, a, c], [b, c, a], [b, c, c], [c, a, b], [c, a, c], [c, b, a], [c, b, c],  
 [c, c, a], [c, c, b]]
```

12

Multinomial and Binomial Coefficients

The Maple functions `multinomial()` and `binomial=()` return what you expect:

```
> binomial(9,4); 9!/(4!*5!);
```

126

126

```
> multinomial(8,3,3,2); 8!/(3!*3!*2!);
```

560

560

```
> expand((x+y+z)^6);
```

```
6 y z5 + 15 y2 z4 + z6 + 6 x y5 + 15 x2 z4 + 6 x5 y + 6 x5 z + 15 x4 y2 + 15 x4 z2 + 6 x z5 + x6
```

$$\begin{aligned}
&+ 60 x^3 y^2 z + 30 x^4 y z + 90 x^2 y^2 z^2 + 60 x^2 y z^3 + 60 x^2 y^3 z + 60 x^3 y z^2 + 30 x y^4 z + 60 x y^3 z^2 \\
&+ 60 x y^2 z^3 + 20 x^3 y^3 + 20 x^3 z^3 + 6 y^5 z + 15 y^4 z^2 + 20 y^3 z^3 + 15 x^2 y^4 + y^6 + 30 x y z^4
\end{aligned}$$

The calculation

```

> multinomial(6,1,3,2); multinomial(6,2,2,2);
          60
          90

```

illustrates $(x_1+x_2+\dots+x_n)^m = \sum(\text{multinomial}(m,k_1\dots k_n) * x_1^{k_1} * x_2^{k_2} * \dots * x_n^{k_n}, k_1+k_2+\dots+k_n = m)$, which you should try to prove either by carefully grouping terms, or perhaps by induction.

r-combinations

We can select r objects, without regard to order, from a set of n objects (distinguishable since elements of a set) in $C(n,r) = P(n,r)/r! = n!/(r!(n-r)!)$ ways, because $P(n,r)$ is the number of ways to choose the r objects in order, and if we disregard the order, we clearly have to divide by $r!$. $C(n,r)$ is called $\text{binomial}(n,r)$ in Maple. It is the number of combinations of n distinct objects taken r at a time.

Let X be a set of cardinality n where $n = n_1 + n_2 + \dots + n_m$ and let $f : X \rightarrow \{1,2,\dots,m\}$. Then f determines an ordered partition of X (that is, a list of nonempty subsets with union X). Suppose the i -th subset, the preimage of i , contains n_i points. How many such functions are there? Well, we can select $C(n,n_1)$ elements to map to 1, then $C(n-n_1,n_2)$ elements to map to 2, and so forth. Thus the number of functions with the preimage of i having cardinality n_i , $i = 1,\dots,m$ is $C(n,n_1)*C(n-n_1,n_2)*\dots*C(n-n_1-\dots-n_{(m-1)},n_m) = P(n; n_1,n_2,\dots,n_m)$; that is, $\text{multinomial}(n,n_1,\dots,n_m)$.

Thus, given n distinct objects and m numbered containers, there are $\text{multinomial}(n,n_1,\dots,n_m)$ ways of placing the objects in the containers, with n_k objects in the k -th container.

Here's another way to look at it: Given n objects where there are n_k objects of type (color, etc.) k , $k=1,\dots,m$, there are $\text{multinomial}(n,n_1,\dots,n_m)$ ways to arrange them in a row (argue by choosing the location).

Here's an example - the number of strings of length 12 which contain 4 a's, 3 b's, 2 c's and 3 d's is

```

> multinomial(12,4,3,2,3);
          277200

```

Note that $\text{binomial}(n,k) = \text{multinomial}(n,k,n-k)$

Partitions of Sets Revisited

Let A be a set of cardinality n .

The number of partitions of A into (nonempty) subsets is $\text{bell}(n)$.

Now consider the number of partitions of A into m subsets A_i , $i = 1, \dots, m$, where A_i has cardinality n_i . Let's order the set of sets A_i by cardinality to obtain a list. The number of such lists is $\text{multinomial}(n, n_1, \dots, n_m)$. If the cardinalities n_1, \dots, n_m are all distinct then each list corresponds to one partition.

If, on the other hand, N_1, \dots, N_k are the distinct values for the cardinalities n_1, \dots, n_m and each N_i occurs p_i times (we call the p_i the multiplicities) then re-arranging the parts of the list corresponding to a given N_i does not change the corresponding partition and so we have to divide by $(p_i)!$ for each i to get the right count. Thus

If N_1, \dots, N_k are distinct, the number of partitions of a set A of cardinality $n = p_1 * N_1 + \dots + p_k * N_k$ into partitions consisting of p_i sets of cardinality N_i , $i=1, \dots, k$, is

$\text{multinomial}(n_1, \dots, n_m) / ((p_1)! \dots (p_k)!)$

where n_1, \dots, n_m is the sequence of N_i 's with each repeated according to multiplicity. Here's a procedure to compute this number

```
> numbparts:=proc(L1,L2)
>   local k,h,C,M,n,p,allcard;
>   if nargs < 2 then
>     return FAIL;
>   else
>     C:=[];
>     M:=[];
>     n:=0; p:=1;
>     for k from 1 to nargs do
>       C:=[op(C),args[k][1]];
>       M:=[op(M),args[k][2]];
>       n:= n + args[k][1] * args[k][2];
>       p:= p * (args[k][2])!
>     od;
>     allcard:=[];
>     for k from 1 to nargs do
```

```

>   for h from 1 to M[k] do
>     allcard:=[op(allcard),C[k]];
>   od;
> od;
> return multinomial(n,op(allcard))/p;
> fi;
> end:

```

Here's an example from our text: The number of ways to divide 43 students into 7 groups such that there are 5 students in each of 2 groups, 6 students in each of 3 groups, 7 students in one group, and 8 students in one group is

```

> numbparts([5,2],[6,3],[7,1],[8,1]);
4609505988517364677515694521600

```

If we allocate each group of students to a named group, and the names are distinct, then we can impose an arbitrary ordering on the names and we are dealing with ordered partitions. Thus the number of ways to assign grades to 43 students so that 8 students get an A, 7 students get an A-, 6 students get B, 6 get C+, 6 get C, 5 get D and 5 get F is

```

> multinomial(43,8,7,6,6,6,5,5);
55314071862208376130188334259200

```

In this context the previous number above is the number of ways 8 students get the same grade, 7 students get the same grade, 6 students get the same grade, 6 students get the same grade, 6 students get the same grade, 5 students get the same grade, and another 5 students get the same grade, and the grades for each group are different, but unspecified.

Problem 3. A small college has 3 dormitories, imaginatively called dorm A, dorm B and dorm C. Each dormitory has a large number of vacancies when 8 new students arrive on campus.

Part (A). How many ways can we distribute the 8 students among the 3 distinguishable dorms if we assume the students are indistinguishable, that is, we just care how many students end up in each dorm?

Part (B). How many ways can we distribute the 8 students among the 3 distinguishable dorms if we assume the students are indistinguishable, that is, we just care how many students end up in each dorm, and if we require that each dorm gets at least one student?

Part (C). How many ways can we distribute the 8 students among the 3 dorms if we regard the students as distinguishable and the dorms as indistinguishable?

Part (D). How many ways can we distribute the 8 students among the 3 dorms if we regard the

students as distinguishable and the dorms as indistinguishable, and we require that each dorm gets at least one student?

Part (E). How many ways can we distribute the 8 students among the 3 dorms if we regard the students as indistinguishable and we regard the dorms as indistinguishable?

Part (F). How many ways can we distribute the 8 students among the 3 dorms if we regard the students as indistinguishable and we regard the dorms as indistinguishable, and we require each dorm gets at least one student?

Part (G). How many ways can we distribute the 8 distinguishable students among the 3 distinguishable dorms?

Part (H). How many ways can we distribute the 8 distinguishable students among the 3 distinguishable dorms if each dorm is to get at least one student?

Part (I). How many ways can we put 3 students in dorm A, 3 students in dorm B, and the remaining 2 in dorm C? Here we assume the students distinguishable, of course.

Part (J). How many ways can we put 3 students in one dorm, another 3 in another dorm and the remaining 2 in the remaining dorm, but we don't care which dorms they end up in? Here we assume the students distinguishable.

Your answers should probably all come from the list

[5, 10, 21, 45, 280, 560, 966, 1094, 5796, 6561],

though maybe not?

>