Math 318 – homework 0 – solutions

Problem 1. Create a two page typeset PDF file with your name and student number near the top of page 1.

Problem 2. Consider arrangements of the letters `ABRACADABRA`?

(a) How many different arrangements are there?
(b) How many of these have the five `A`s at the start?
(c) How many of these have the five `A`s all together?
(d) How many of these have no two consecutive `A`s? (Hint: if the positions of the `A`s are $i < j < k < \ell < m$, then $i, j-1, k-2, \ell-3, m-4$ are distinct numebrs if and only if there are no consecutive `A`s.)

Solution.

(a) The number of repetitions of letters are $5, 2, 2, 1, 1$, so the number of arrangements is $\binom{11}{5,2,2,1,1} = \frac{11!}{5!2!2!1!1!}$.
(b) If all the As are at the start, there are $\binom{6}{2,2,1,1} = 180$ arrangements of the remaining letters.
(c) There are 7 options for the location of the block of As, and 180 arrangements of the remaining letters in each of these, so the total is $7 \cdot 180$.
(d) The positions of the As must be 5 numbers from $\{1, \dots, 11\}$ with no two consecutive numbers. As the hint implies, this is equivalent to $i, j-1, k-2, \ell-3, m-4$ having no repetitions, so $i, j-1, k-2, \ell-3, m-4$ are any five numbers from $\{1, \dots, 7\}$. Therefore there are $\binom{7}{5} = 21$ possible sets of locations for the As. For each of these, there are 180 arrangements of the remaining letters, as above, so the overall answer is $21 \cdot 180$.

Problem 3. Write a python function that takes a string and returns the number of distinct arrangements of its letters. You may assume the string only includes lowercase letters. (`numpy.math.factorial(n)` can be used to calculate factorials.)

Solution. Sample code:

```python
import numpy as np
from math import factorial

### assuming only lowercase letters are used:

def arrangements(word):
    N = factorial(len(word))
    for x in 'abcdefghijklmnopqrstuvwxyz':
        N /= factorial(word.count(x))
        return(N)

### for general strings:

def arrangements(word):
    S = set(word) # the set of characters appearing
    N = factorial(len(word))
    for x in S:
        N /= factorial(word.count(x))
        return(N)
```