Assignment 4, MACM 204, Fall 2014

Due Friday November 7th at 4:00pm.
Late penalty: -20% for up to 72 hours late. 0 after that.

Michael Monagan.

Question 1

The Maple command `sum( f(i), i=a..b )` computes a formula for the $\sum_{i=a}^{b} f(i)$.

For example, the sum of the first $n-1$ integers is $\sum_{i=0}^{n-1} i$ is given by.

```
> sum( i, i=0..n-1 );
```

$$\frac{1}{2} n^2 - \frac{1}{2} n$$

Calculate **and simplify** the following 6 sums

(i) $\sum_{i=0}^{n} i^2$ (ii) $\sum_{k=0}^{n} k \cdot \binom{n}{k}$ (iii) $\sum_{k=1}^{\infty} \frac{k^2}{2^k}$ (iv) $\sum_{i=0}^{n} a^i$ (v) $\sum_{i=1}^{n-1} \sum_{j=1}^{i} j \cdot (n-j)$ (vi) $\sum_{i=0}^{\infty} \frac{x^i}{i!}$

Question 2

Write a Maple procedure CountMax takes as input a list of $n > 0$ values and returns the number of times the maxium element in the list appears. For example

```
> L := [1,3,5,1,5,1];
```

$$L := [1, 3, 5, 1, 5, 1]$$

```
> CountMax(L);
```

$$2$$

Question 3

This question is related to Newton's law of cooling.
Let $T(t)$ be the temperature of a body of liquid at time t. Let $Troom$ be the room (ambient) temperature of the surrounding medium (air). The DE is $T'(t) = k \cdot (Troom - T(t))$ where k is the cooling rate constant.
Solve the differential equation in Maple for Troom = 20 degrees and an initial temperature of 40 degrees.
Given also that $T(20) = 30$, determine k. Now compute T(60). Do all the calculations in Maple.
Finally graph $T(t)$ for $0 \le t \le 60$ together with the room temperature on a suitable domain/range .

Question 4

Carbon 14 decays into Nitrogen 14. Using Google, find the half life H of Carbon 14.
The differential equation modeling radioactive decay is
$y'(t) = -k \cdot y(t)$
where k is the decay constant and y(0) is the initial concentration of Carbon 14. Given the half life is H, that is, given that $y(H) = \frac{y(0)}{2}$, determine k. You can do this one by hand at first but then do it in Maple.
Solve the DE in Maple and graph the solution for $y(0) = 1$ on a suitable domain.

Question 5

Suppose we have a 400 liter tank. Suppose 8 litres per minute of salt water (brine) flows into the tank at the top and then flows out of the tank at the bottom. Assume for simplicity that the salt water in the tank is stirred so that its concentration is uniform in the tank. Let $S(t)$ be the amount of salt, in grams, in the tank at time t minutes.

Suppose the salt water flowing into the tank has concentration 100 grams per liter.

Find the differential equation to model the change in S(t).

Assuming there is no salt in the tank at time t=0 solve the differential equation using Maple.

What is S(∞)? That is, how much salt is in the tank after a long time?

Now graph S(t) for a suitable domain.

Solution 5

Question 6

The logistic growth with harvesting model for a population $y(t)$ at time t is given by

$$y'(t) = \boldsymbol{a} \cdot y(t) \cdot \left(Y_{\max} - y(t)\right) - H$$

Here *Ymax* is the maximum sustainable population of the environment, **a** is a constant and H is a constant harvesting rate. For Ymax = 8000, a=0.0001, and H = 1000, using the DEplot command, graph y(t) for $0 \le t \le 10$ for the initial values $y(0)$ in 1000, 5000, 8000 and 10000.

Now determine populations y for which $y' = 0$, i.e., find the initial polulations for which there is no growth or decline. You should get two. Graph these on the same graph - you should get two straight lines.

Question 7

Using the **disk** command in the **plottools** package to create a solid disk, create an animation which simulates the earth (a blue disk) revolving around the sun (a yellow disk) in an eliptical orbit. To get an animation, generate n frames of the animation $Frames := [F_1, F_2, \ldots, F_n]$ where each frame is a plot and display them together using the display command as follows

```
> display( Frames, insequence=true, scaling=constrained );
```

Question 8

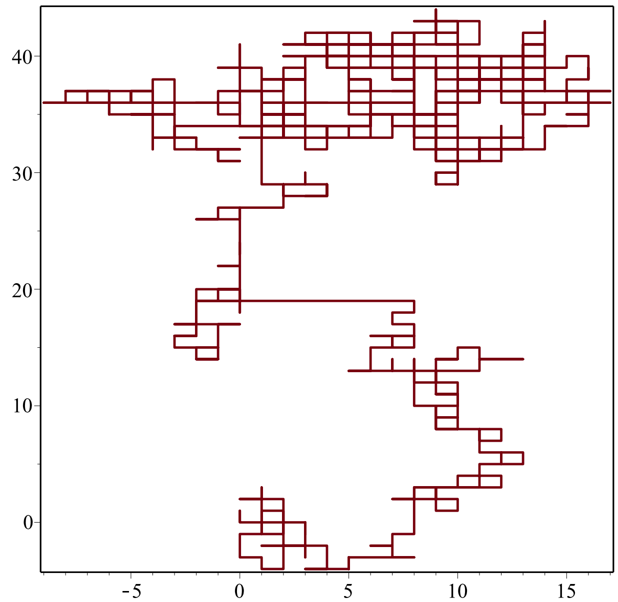
The following is a solution to the random walk exercise in Assignment 3 by a student. It is correct but very slow for large values of n.

```
> R := rand(1..4):
  Walk := proc(n) local x,y,i,P,Q;
    x := 0;
    y := 0;
    Q := [[0,0]];
    for i from 1 to n do
        P := R();
        if P=1 then x := x+1;
        elif P=2 then x := x-1;
        elif P=3 then y := y+1;
        else y := y-1;
        fi;
        Q := [op(Q),[x,y]];
    od;
```

```
    Q;
  end:
> Walk(3);
```

$$[[0,0],[1,0],[2,0],[1,0]]$$

```
> plot( Walk(1000), style=line, axes=box );
```



To measure the efficiency of the Maple procedure we need to time it for different values of *n*. To time it use the Maple time(); command as follows

```
> start := time(); Walk(n): timetaken := time()-start;
```

Input the above program and time how long this code takes to execute for n=4000, n=8000, n=16000, n=32000.

Interpret the timings you get. Is the running time of the Walk procedure linear in n, quadratic in n, cubic in n, or something else?

The problem is the code `Q := [op(Q),[a,b]]` which appends the next point to the list . Doing this in a loop means that we are creating a list of length 2 then 3 then 4 then 5 and so on. Creating lists of length 1000 takes time. This makes the procedure slow. One solution is to store the points in an Array instead of a list. To create an Array of n values indexed from 1 to n use

```
> A := Array(1..n);
```

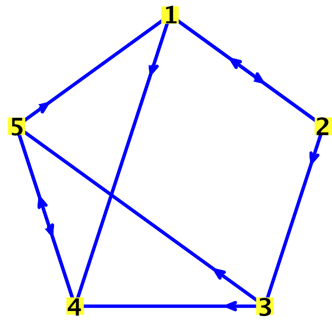
To insert a value into the Array use

```
> A[1] := [0,0];
```

Inserting a value into an array is fast even if the array is very large. Since Maple's plot(A,style=line) command accepts for A either a list of points or an Array of points, we don't need to change anything else

Now reprogram the Walk procedure to use an Array and retime the improved code for n=4000, n=8000, n=16000, n=32000, n=64000. It should make a huge difference.

Question 9



Shown above is a directed graph with 5 vertices. The graph represents 5 web pages and the directed edges represent the hyperlinks between the web pages. Input and draw the graph in Maple (see ?GraphTheory and use the ?Graph and ?DrawGraph commands).

Now, starting at vertex 1, simulate a random walk of length $N = 20,000$ steps through the graph. Store the walk in an array W. So since vertex 1 is connected to vertex 2 and 4, the random walk will go to vertex 2 or 4 with probability 0.5. So if $W_1 = 4$ this means after the first random step you are on page (vertex) 4. At the end of the random walk, for each vertex, calculate the probability the walker visits that vertex.

Now modify your code to do the following. After each 100 steps, for each vertex, calculate the probability that the walker was on that vertex. For each vertex, this generates a sequence of 200 probabilities that should converge.

Graph these 5 sequences on a single graph. Include a legend. You should get a picture like the following (Page 1 corresponds to vertex 1)