```
> #Ok to post Homework
> #Jeton Hida, Assignment 24, November 29, 2021

> #Question 1

> #For the attendance quiz I made silly mistakes, the first one I
  mistakenly tried to treat it all as if velocity was x(t) instead of
  x'(t), I knew accelaration is the derivative of velocity, so I
  treated that as x'(t) instead. Basically, it sums up to me being
  one derivative off for the whole problem, and that in turn caused
  me to get it wrong. For number 2 on the attendance quiz, I was
  unsure where to go with the info given, specifically the part where
  we equate ma = -mg -> m(x''(t)) = -mg and we cancel m's. This was
  the only issue, I see it now, every other step here was easy to do,
  if I only got that first step I would've solved this easily, but
  unfortunately I didn't.

> #Question 2

> #rate of change of the rate of change of acceleration, acceleration
  is x''(t) rate of change of that is x'''(t) rate of change of that
  which is x''''(t). We are given the rate of change of this one,
  (5th derivative), so x'''''(t)=120m/s^3 (homework says sec^3, but i
  believe it should be to the 5th power so, sec^5) We are going to
  perform a series of integrations, with solving for C at each step
  from given initial conditions. At the end once we find x(t) we will
  plug in t=2 to the equation.

> #x'''''(t) = 120, x(0) = 0; x'(0) = 0; x''(0) = 0; x'''(0) = 0;
  x''''(0) = 0
  #x''''(t) = 120t + C -> 0 = 120(0) + C, C=0, x''''(t) = 120t
  #x'''(t) =60t^2 + C -> 0 = 60(0^2) + C, C=0, x'''(t) = 60t^2
  #x''(t) = 20t^3 + C -> 0 = 20(0^3) + C, C=0, x''(t) = 20t^3
  #x'(t) = 5t^4 + C -> 0 = 5(0^4) + C, C=0, x'(t) = 5t^4
  #x(t) = t^5 + C -> 0 = (0^5) + C, C=0, x(t) = t^5

  #x(2) = 2^5 = 32, Hence the particle is at a distance 32 meters
  from the starting point after 2 seconds!

> #Question 3 i.

> #ma = -mg, m(x''(t))=-mg, m cancels out x''(t) = -g, g=10. x'(0)=0,
  x(0)=100m Now do a series of integration and solving for C based on
  initial conditions!
  #x''(t) = -10
  #x'(t) = -10t + C -> 0 = -10(0) + C, C=0, x'(t) = -10t
  #x(t) = -5t^2 + C -> 100 = -5(0^2) + C, C=100, x(t)=-5t^2+100
  #Ball is on the ground when x(t) = 0, so let's solve for t to find
  how many seconds it will take to reach ground
  #-5t^2 + 100 = 0, -5t^2 = -100, t^2 = 20, t = ± sqrt(20), a
  negative value for time makes no sense so the ball reaches the
  ground when the time is t=sqrt(20)

> evalf(sqrt(20))
                                4.472135954                                (1)

> #The ball hits the ground at t=4.472135954 seconds

> #ii.
```

```
> #Ball is dropped from height of 100 meters, x(0) = 100, air-
  resistance is equal to 2mv where v is velocity, velocity is also
  equal to x'(t) here. Acceleration is x''(t) = -10 where 10 = g in
  units of m/s^2. Setting up an equation we have this.
```

```
> #mx''(t) = -10m - 2m(x'(t)), the value 2m(x'(t)) is negative here
  as the air resistance value is related to the velocity. Velocity of
  this ball will also be negative for the same reason the
  acceleration is negative, because of the direction the ball is
  moving. All the m's will cancel out. With that said we are left
  with the following equation,
```

```
> #x''(t) = -10 - 2x'(t)
```

```
> dsolve({D(D(x))(t)=-10-2*D(x)(t),x(0) = 100,D(x)(0)=0},x(t))
```

$$x(t) = -\frac{5\,e^{-2\,t}}{2} - 5\,t + \frac{205}{2} \quad \textbf{(2)}$$

```
> evalf(solve((-5/2)*exp(-2*t)-5*t+(205/2)=0,t))
```

$$20.50000000, -1.90112928 \quad \textbf{(3)}$$

```
> #The ball will reach the ground at 20.5 seconds
```

```
> #Question 4
```

```
> #(a)
> #x(n) = x(n-1), this is the format for a discrete time system
  specifically a first-order discrete system. The general format is
  #x(n) = f(x(n-1),x(n-2),...,x(n-k)) for a k-th order discrete
  dynamical system. Where we say the value of our quantity (x) at a
  time step n is equal to a function containing the value of our
  quantity (x) at previous time step(s).
```

```
> #(b)
  #The orbit of a discrete time dynamical system with one quantity
  starting at x(n) = x0 up to n = K.
  #What we mean by this is that with our discrete time dynamical
  system which we have said the value x(n) is equal to a function of
  previous time steps f(x(n-1),x(n-2),...,x(n-k)). We start an
  initial value we call x0. To find out where we are at the next
  timestep we plug this point x0 into our function of our quantity.
  This next point we land at we call x1. An orbit pertains to the
  idea that after several iterations, up to n = K, where does our
  quantity end up at? An orbit means we are actually "orbitiing"
  around several points. In this case, after K timesteps we notice
  that our quantity only seems to take on several values and nothing
  else. At timestep K-3 we take on a value of A, we'll call it. Then
  at K-2 we take on a value of B, and then at K-1 we go back to A,
  etc. We continue to repeat this "orbit" around these two values A
  and B.
```

```
> #(c)
> #An equilibrium solution for a discrete time dynamical system
  refers to a point that we start at, and never leave. If our
  quantity x(n) has an equilibrium solution we'll call x*, then if we
  start at x* we will never leave x*. The point in the next timestep,
  will be x* and the point in K timesteps will always be x*. A good
  way to think about this is for discrete time systems that model
  population growth between generations. An easy equilibrium point to
  understand is 0. If we start with our population equalling to 0, we
```

will never grow from 0, how can we have more members of a population being born when we have none to begin with, we have no members able to give birth to new life. The system will remain at the equilibrium point 0.

```
> #(d)
```

> #A stable equilibrium solution is an equilibrium solution to a discrete time dynamical system with one other quality to it. It is true like before that if we start at this value, we remain at this value even going forward K timesteps. The difference here is that a stable equilibrium solution does not require you to start at it to eventually reach it. If you start at a point near the stable equilibrium, your value of x(n) will eventually reach a value close to the stable equilibrium and then stay at it indefinitely. Say we have a stable equilibrium value of .5, and we begin at .25, after K timesteps (maybe less depending on the function) we will end up near this stable equilibrium value of .5 and stay at it.

```
> #Question 5
```

```
> #(a)
```

> #To find the stable equilibria (stable fixed point) using orbits, of a discrete time dynamical system we can use the Maple procedure by Dr. Z Orb. Which really takes a function f(x(n-1),...,x(n-k)) and using our initial conditions say x(0)=x0. Iterates it K timesteps ahead. The idea is that after many iterations we will notice a pattern of what the value of x(n) is. If we start at x0 and stay at x0 after several timesteps then we know that x0 is a fixed point. If we start near x0, but not exactly at it and we still end up at x0 after K timesteps, we know x0 is a stable fixed point. If we do not end up at x0 after starting near it, then we can say x0 is an unstable fixed point.

```
> #(b)
```

> #To algebraically find a set of fixed points for a function of a discrete time dynamical system we have a few steps. We take a x(n)= f(x(n-1)) and rewrite the underlying function of f(x(n-1) to f(x). To find fixed points we now equate x to f(x) so x=f(x) and find all values of x that satisfy this. These values of x are in our set of fixed points for the specific discrete time dynamical system.

```
> #(c)
```

> #To find out if these fixed points we found are indeed stable, we now use calculus. We take our function f(x) and derive it once. We are now left with f'(x). We now substitute our fixed point values one at a time for x and from there we see the value of our f' at this equilibrium point. The absolute value of f' at this equilibrium point tells us whether or not this point is stable. If the absolute value of f' at the point is < 1 then we can say our equilibrium point is stable, if infact our equilibrium point at f' is > 1 then it is infact unstable. If in the case it is = 1 then we cannot conclude whether it is stable or unstable just from this calculus method alone, may need to infact use our Orb procedure and figure it out numerically.

```
> #(d)
> read "/Users/jeton/Desktop/Math 336/DMB.txt"
```

First Written: Nov. 2021

This is DMB.txt, A Maple package to explore Dynamical models in Biology (both discrete and continuous)
accompanying the class Dynamical Models in Biology, Rutgers University. Taught by Dr. Z. (Doron Zeilbeger)

The most current version is available on WWW at:
http://sites.math.rutgers.edu/~zeilberg/tokhniot/DMB.txt .
Please report all bugs to: DoronZeil at gmail dot com .

For general help, and a list of the MAIN functions,
type "Help();". For specific help type "Help(procedure_name);"

For a list of the supporting functions type: Help1();
For help with any of them type: Help(ProcedureName);

For a list of the functions that give examples of Discrete-time dynamical systems (some famous), type: HelpDDM();
For help with any of them type: Help(ProcedureName);

For a list of the functions continuous-time dynamical systems (some famous) type: HelpCDM();
For help with any of them type: Help(ProcedureName);

------------------------------ **(4)**

```
> Help(Orb)
```

Orb(F,x,x0,K1,K2): Inputs a transformation F in the list of variables x with initial point pt, outputs the trajectory of
of the discrete dynamical system (i.e. solutions of the difference equation): x(n)=F(x(n-1)) with x(0)=x0 from n=K1 to n=K2.
For the full trajectory (from n=0 to n=K2), use K1=0. Try:
*Orb(5/2*x*(1-x),[x], [0.5], 1000,1010);*
*Orb([(1+x+y)/(2+x+y),(6+x+y)/(2+4*x+5*y)],[x,y], [2.,3.], 1000,1010);* **(5)**

```
> Orb([(x+1)/(x+2)],[x],[1.0],1000,1010)
```

[[0.6180339888], [0.6180339888], [0.6180339888], [0.6180339888], [0.6180339888], **(6)**
[0.6180339888], [0.6180339888], [0.6180339888], [0.6180339888], [0.6180339888],

[0.6180339888], [0.6180339888]]

```
> A:=evalf(solve(x=(x+1)/(x+2),x))[1]
```

$$A := 0.6180339880 \quad (7)$$

```
> B:=evalf(solve(x=(x+1)/(x+2),x))[2]
```

$$B := -1.618033988 \quad (8)$$

```
> C:=diff((x+1)/(x+2),x)
```

$$C := \frac{1}{x+2} - \frac{x+1}{(x+2)^2} \quad (9)$$

```
> subs(x=A,C)
```

$$0.1458980339 \quad (10)$$

```
> #This value is < 1 so .6180339880 is a stable equilibrium point,
  which agrees with our Orb procedure.
```

```
> subs(x=B,C)
```

$$6.854101940 \quad (11)$$

```
> #This value is > 1 so -1.618033988 is an unstable equilibrium
  point.
```

```
> #ii.
> Orb([5/2*x*(1-x)],[x],[.5],1000,1010)
```

[[0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000], [0.6000000000]] (12)

```
> evalf(solve(x=5/2*x*(1-x),x))
```

$$0., 0.6000000000 \quad (13)$$

```
> diff(5/2*x*(1-x),x)
```

$$\frac{5}{2} - 5x \quad (14)$$

```
> subs(x=0,%)
```

$$\frac{5}{2} \quad (15)$$

```
> #This value is > 1 so 0 is an unstable equilibrium point.
```

```
> subs(x=.6,5/2-5*x)
```

$$-0.500000000 \quad (16)$$

```
> #The absolute value of this is < 1 so .6 is a stable equilibrium
  point, and agrees with Orb
```

```
> #iii.
> Orb([7/2*x*(1-x)],[x],[.5],1000,1010)
```

[[0.5008842111], [0.8749972637], [0.3828196827], [0.8269407062], [0.5008842111], [0.8749972637], [0.3828196827], [0.8269407062], [0.5008842111], [0.8749972637], [0.3828196827], [0.8269407062]] (17)

```
> evalf(solve(x=7/2*x*(1-x),x))
```

$$0., 0.7142857143 \quad (18)$$

```
> diff(7/2*x*(1-x),x)
```

$$\frac{7}{2} - 7x \quad (19)$$

```
> subs(x=0,7/2-7*x)
```

$$\frac{7}{2} \quad (20)$$

```
> #This value is > 1 so 0 is an unstable equilibrium point.
```

```
> subs(x=.7142857143,7/2-7*x)
```

$$-1.500000000 \quad (21)$$

```
> #This absolute value is also > 1 so .7142857143 is also an unstable
  equilibrium point. These both agree with Orb as there is an orbit
  of period 3 for this function f(x).
```