1. A farmer owns 1,000 acres of farmland. Her options are to breed cattle or to plant wheat, corn, or tomatoes. It takes four acres to support one head of cattle. Annually, 12,000 hours of labor are available. The table below provides information regarding the profit, yield, and labor needs for the four economic activities.

	Cattle	Wheat	Corn	Tomatoes
Profit	\$1600/head	\$5/bushel	\$6/bushel	\$0.5/lb
Yield per acre	0.25 head	50 bushels	80 bushels	1000 lb
Annual labor requirement	40 hrs/head	10 hrs/acre	12 hrs/acre	25 hrs/acre

   Furthermore, it is required that at least 20% of the farmland that is **cultivated** must be used for cattle breeding, at most 30% of the available farmland can be used for growing tomatoes, and the ratio between the amount of farmland assigned to growing wheat and that left **uncultivated** should not exceed 2 to 1.

   Using optimization, determine how the farmer should allocate her land in order to maximize profit.

2. You have an inventory of 120 20-ft rods, 160 15-ft rods, and 40 8-ft rods. Your customer requires 200 10-ft rods, and 250 6-ft rods. To satisfy their demand, you need to cut your existing rods. Using optimization, determine the solution that minimizes waste (total length of rods that needs to be discarded after cutting).

3. Solve the wave equation

$$\frac{\partial^2 u(t,x)}{\partial t^2} = \frac{\partial^2 u(t,x)}{\partial x^2}$$

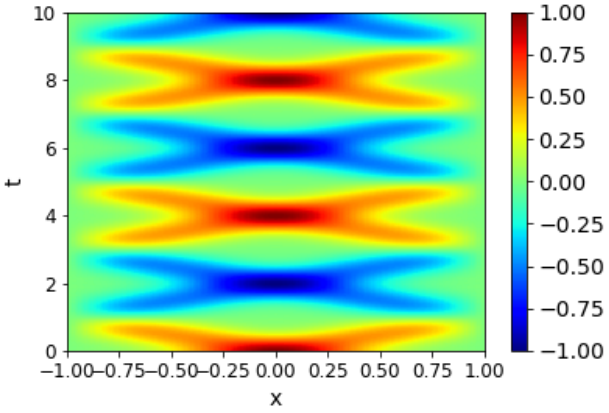
   subject to the boundary conditions $u(t,-1) = 0$ and $u(t,1) = 0$ and the initial conditions:

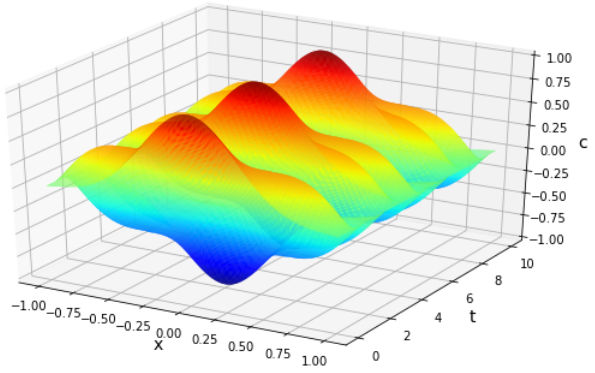
$$u(0,x) = \exp\left(-10x^2\right)$$

   and

$$\frac{\partial u(0,x)}{\partial t} = 0.$$

   Your code should generate the following plots





Hint: You need to convert the second-order time derivative to two coupled first-order differential equations.

4. Construct a Class that implements a Vigenere Cipher.

   The following commands

```
# define the alphabet
alphabet = 'abcdefghijklmnopqrstuvwxyz'
alphabet += alphabet.upper() + '.,!?' + ' '

mycipher = Cipher(alphabet)

message = 'This is my secret message!?!'
```

```
key = 'my key'

print message
print

encoded_message = mycipher.encode(key,message)
print encoded_message

decoded_message = mycipher.decode(key,encoded_message)
print decoded_message

print

key = 'longer key'
encoded_message = mycipher.encode(key,message)
print encoded_message

decoded_message = mycipher.decode(key,encoded_message)
print decoded_message
```

should yield the output

```
This is my secret message!?!

aFhCdGExlIdQqAqoxxyCrCeEqv!h
This is my secret message!?!

 vvydzrjqWkGrivvsjqCDGnmio!h
This is my secret message!?!
```

For background on the Vigenere Cipher, please see the Wikipedia articles:

`https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher`
`https://en.wikipedia.org/wiki/Tabula_recta`

and the attached handout (will be posted soon).

Using modular arithmetic will greatly simplify your code. To understand modular arithmetic, evaluate the output of the following code:

```
for i in range(10):
```

```
j = i % 4
k = (i+2) % 4
print i, j, k
```