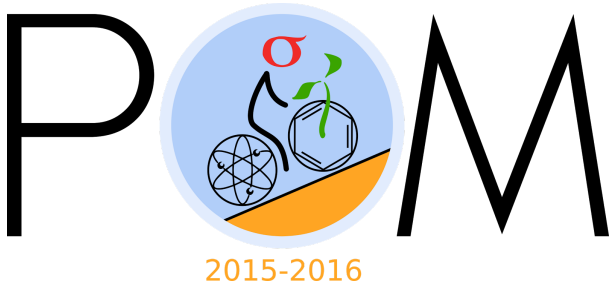
PROBLEM OF THE MONTH



January, 2016

MATHEMATICS

5 points:

Find the number of digits in the number 2^{100001}. You may use a basic calculator, but not the logarithmic function. Describe your procedure.

Answer: 30103

Hint: note that $2^{10} = 1024$

Solution:

Since $2^{10} = 1024$, we can first approximate it as 1000. Under such an approximation, every 10 factors of 2 give additional 3 digits to the number. We can now improve the accuracy by noting that $2^{10n} = 10^{3n} 1.024^{n}$. Let us use calculator to find the number n such that the correction 1.024^{n} is closest to 10. This turns out to be n=97 , and $1.024^{n} \approx 9.97920$. Therefore, there is an additional digit added each time when n passes the number divisible by 97.

$2^{100001} = 2 \times 2^{10n}$, where n=10000. by dividing 10000 by 97, we find that there are 103 such "bonus" digits, in addition to 3n=30000, that would be expected if 2^{10} were replaced with 1000.

Therefore, the total number of digits is 30103.

10 points:

Find the number of digits in the number $125^{1,000,000}$. You may use a basic calculator, but not the logarithmic function. Describe your procedure.

Answer: 2,096,910

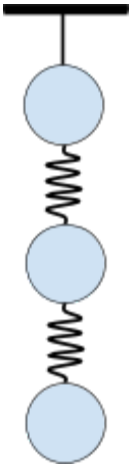
Hint: note that $125 = 1000/2^3$.

Solution:

We note that $125 = 1000/2^3$. Therefore, in order to solve the problem we need to count the number of digits in number 2^{10n}, n=300,000.

As shown in solution of 5 pt. problem, in the simplest approximation it is 3n=900,000. In order to find the first correction, we calculate n/97=3092.78… This means that there are additional 3092 digits in the number 2^{10n}. However, since n is so large we no longer can assume that $1.024^{97} \approx 9.97920$ is equal to 10. Instead, we need to take additional correction, i.e. to find a number k such that $.997920^k$ is close to 0.1. This corresponds to k=1105. Each time when k passes a value divisible by that than number, we should "skip" one digit. There are 2 such occurrences for n=3092. We conclude that the number $2^{3,000,000}$ has (900000+3092-2) digits, and therefore $125^{1,000,000} = (10/2)^{3,000,000}$ has 3,000,000-(900000+3092-2)= 2,096,910 digits

PHYSICS



5 points: Three identical balls are connected by identical weightless springs and suspended on the string (see figure). Find the accelerations of the balls at the moment the string is burnt.

Answer: 3g, 0, 0 from top to bottom

Hint: The tensions of springs do not change immediately after the string is burnt.

Solution: Let us consider the system before the string is burnt. All three balls are in equilibrium, that is, the net force on each of them is zero. Right after the string is burnt the lengths of springs do not change immediately and neither do their tensions. This means that the middle and the bottom balls will have zero acceleration in the first moment. Let us now consider the top ball. The tension of the string before it is burnt is $3mg$. This means that right after the string is burnt the net force acting on the top ball is equal to $3mg$ and is directed downwards. We conclude that the acceleration of the top ball in the first moment is equal to 3g.

Remark 1: it helps to think about free body diagrams for each ball just before and just after the string is burnt.
Remark 2: Let us consider the motion of the center of mass of the system. After the string is burnt the net force acting on the system is $3mg$ and the net mass of the system is $3m$. This means that the center of mass should accelerate with acceleration g. However, because the acceleration of bottom and middle balls are zero, the top ball should accelerate with the acceleration $3g$.

10 points

Superman jumps from a bridge of height H, falls vertically and lands on a cart of mass M moving with velocity V. Find the maximal amount of heat released during the collision of Superman with the cart if the mass of Superman is m.

Hint: Use the conservation of the horizontal component of momentum and the conservation of energy.

Answer: $Q = mgH + \frac{mM}{m+M}\frac{V^2}{2}$

Solution: Let us consider the conservation of the horizontal component of momentum. We have $MV = (m+M)v$, where v is the horizontal velocity of both cart and superman some time after superman's landing. Now we write the conservation of energy $mgH + \frac{MV^2}{2} = \frac{(m+M)v^2}{2} + Q$. Assuming that the entire difference between mechanical energies is released in the form of heat (that is, that the released heat is the maximal possible), we find $Q = mgH + \frac{MV^2}{2} - \frac{(m+M)v^2}{2} = mgH + \frac{mM}{m+M}\frac{V^2}{2}$.

CHEMISTRY

5 points

When Alice, a chemistry teacher, came to her lab, she saw Bob, her technician, staring at the glass flask with some liquid. “Look, Alice. Don’t you find it odd?” - Bob said.
“What exactly?” Alice replied.
“The flask. You asked me to prepare 3% hydrogen peroxide solution. I’ve done that. And it started to bubble. It started to bubble fifteen minutes ago, and I have no idea when it is going to stop.”
“What did you do with it, Bob?”, Alice asked.
“Nothing special, Alice. I just took 900 mL of distilled water, added 100 mL of 30% hydrogen peroxide solution to it, and stirred the liquid with a glass stick.”
“Bob, what happened with your finger? Why have you bandaged it?”
“That is just a minor cut, Alice. The glass stick had a sharp edge, and it cut my finger. The amount of blood was small, I even didn’t have to interrupt my work: I finished stirring, and stepped out to bandage my finger. By the way, that is why I didn’t see the moment when the bubbling started: when I came back, the solution has already been bubbling.”
“Now I see what is the reason, Bob. I am afraid, you have to prepare a fresh solution, and to clean the glassware carefully before that. The reason why the bubbles form is …”
Can you continue Alice’s explanations?

Hint: Bubbling (gas formation) means some reaction occurs. However, to initiate a reaction, Bob had to add some reactant (and, since the bubbling is intense, the amount of the added reactant should be significant), or to heat the flask, or to irradiate it with a bright light, or to do something of that kind. However, it seems he hasn’t done anything. Or he has? Frankly, it cannot be ruled out that he accidentally dropped something into the flask (just a tiny drop he haven’t noticed). What could it be, and why could it cause bubbling?

Answer:

Any chemical reaction obeys *stoichiometry* rules, which means that in a reaction:
$A + B \rightarrow C$
a certain amount of molecules A is needed to react with one molecule B. When all molecules A or B have been consumed, the reaction stops. However, in our case, evolution of gas (which is a product of decomposition of hydrogen peroxide) continues for a very long time, and, based on Alice’s words, we can conclude it will continue until all hydrogen peroxide is consumed. That seems odd, because Bob didn’t add anything to this flask intentionally, and even if he dropped something into the flask by accident, the amount of this material was tiny. How can that be possible?
To answer this question, one has to remember that some type of chemicals, in contrast to other reactants, are not consumed during chemical reactions. These chemicals are called *catalysts.* They facilitate a reaction without being consumed during it. One type of organic catalysts called

enzymes are found in our body that catalyze various chemical reactions to support our metabolism. Among those enzymes, an enzyme *catalase* is responsible for accelerating the reaction of decomposition of hydrogen peroxide. It is a very important enzyme, because every cell of our body consumes oxygen and uses it for aspiration (to burn glucose into carbon dioxide, a process that is the main source of energy in our body). During this process, some amount of hydrogen peroxide is formed as a byproduct. Hydrogen peroxide is a very active compound that may damage our DNA or proteins, and that is why each cell of our body produces a special enzyme, catalase, to decompose it to water and oxygen.
That is why our tissues, and especially our blood is a good source of catalase. One drop of our blood is sufficient to start a reaction of recomposition in the hydrogen peroxide solution, and this reaction will last until all H_2O_2 is decomposed.
Alise correctly concluded that when Bob cut his finger, the glass stick became contaminated with Bob's blood, and when Bob started to stir the solution in the flask, he launched the reaction of decomposition hydrogen peroxide. Since the catalyst (catalase) is already there, it is impossible to stop the reaction, and the only option is to prepare a fresh solution in a clean glassware.

10 points:

You are again playing the "Escape the Room" game. You found a secret safe mounted to the wall. The safe is locked, and the lock seems to be controlled electronically. You found that turning off the light in the room deactivates the lock, so you can open the safe when the light is off. Unfortunately, the door of the safe seems to be connected to some switch that does not allow you to turn the light on when the door is open. In other words:

- when the light is on - the safe is locked;
- turning the light off unlocks the safe;
- opening the safe's door blocks the light switch, so you cannot turn the light on until the safe's door is closed;
- turning the light on locks the safe.

There is no windows in the room, and when you turn the light off the room is absolutely dark. When you examined the opened safe by touch you found nothing in it. Your hypothesis is there is probably some message written on the safe's internal surface. Unfortunately, you cannot check this idea, because, according to the rules of this game, all players are supposed to leave all electronic gadgets (smartphones, flashlights etc) outside.
The complete list of artefacts and materials in this room is as follows:

- The book "How to win friends and influence people" by Dale Carnegie.
- The magnifying glass;
- Two portraits of Sherlock Holmes;
- Three used smoking pipes;
- A box of Havana cigars;
- An empty box of matches;
- A flashlight with completely empty alkaline batteries;
- Pliers, a knife, and scissors;

- Porcelain dishes and cups;
- A bottle of mineral water;
- Two 1 L glass flasks with transparent and colourless liquids; the flasks are ⅓ full, the labels on the flasks say: “1% luminol solution”, and “3% sodium hydroxide solution”;
- A 200 mL bottle of standard 3% hydrogen peroxide solution;
- The book “Cat’s cradle” by Kurt Vonnegut; the book is opened on the page 33;
- There is also the number “221b” on the wall.

Assuming your guess about the secret message was correct, how can you obtain the code hidden in the safe?

Hint: Since it is not possible to see anything in the absence of light, you have to find a way to produce it. Your friend (he is the guy with whom you are playing this game; he took Latin classes in school, so he knows Latin a little bit) told you that the name “luminol” resembles Latin “lux” (which means “light”). Maybe, this is a clue? Indeed, immediately upon having learned about that, you remembered that luminol, when oxidized by hydrogen peroxide in a basic media, produces light. But you also remember some catalyst is needed for that. What substance can serve as a catalyst? You cannot remember that. You are waving a flashlight with completely empty batteries (one of the things you found in the room), and mumbling: “Alice, Bob. Alice, Bob…” Why have these two names come to your mind? Why are you waving the flashlight? You yourself cannot explain that…..

Answer:

As a rule, oxidation of all compounds generates energy in a form of heat. Luminol is a rare exception: when it is being oxidized, the energy that is formed during that process is released in a form of light. If we mix a luminol solution with an oxidizer (one of the best oxidizer are activated oxygen molecules) in the presence of alkali, the solution starts to glow. The only question is how to generate activated oxygen molecules. The best source of activated oxygen in hydrogen peroxide. When hydrogen peroxide is mixed with some catalyst that catalyzes its decomposition, activated oxygen is formed first. In these young oxygen molecules covalent bonds between oxygen atoms are not completely formed yet, so they are very active, and they can oxidize luminol to produce light.

What can catalyze decomposition of hydrogen peroxide? Actually, many substances can do that. Firstly, most *transition metal* salts can do that, including iron, manganese, nickel, cobalt salts. One possible source of manganese containing compounds are dead alkaline batteries. In the room, you have a flashlight with dead alkaline batteries, a knife, and pliers. You can simply break the dead battery and drop the content into the mixture of luminol, sodium hydroxide and hydrogen peroxide solutions.

Another option is to use your own blood as a catalyst (see the 5pt problem). By the way, the alkaline solution of luminol/hydrogen peroxide is used by criminalists to detect the traces of blood: when you treat the surface that is contaminated with blood with this solution, the blood traces start to glow.

BIOLOGY

5 points:

Question: Mammalians and birds are homeotherms, which means they maintain the temperature of their bodies constant. In contrast, the majority of other animals have no special mechanisms that allow them to maintain temperature constant. Nevertheless, some species, including insects, are exceptions to that rule. Which insects are capable of maintaining body temperature at the level that is considerably higher or lower than that the ambient temperature, and which mechanisms do they use?

Solution:

Freezing temperatures are detrimental to many forms of life, including most insects. Insect are exothermic (cold-blooded), which means they cannot produce their own body heat. So to survive and thrive in climates such as ours, insects have developed several ways to deal with cold weather.

The first strategy is to avoid freezing conditions altogether. The classic example of this is the monarch butterfly, which migrates south in the fall to overwintering sites in Mexico. In the spring, the monarch population makes its way back north. Eventually the children or grandchildren of last year's monarchs return to Michigan. Pest insects such as armyworms, earworms, potato leafhoppers, and some grain aphids do not survive the winter in Michigan either. Instead, populations continuously reproduce in southern states, and insects move north with spring weather fronts to recolonize northern states. The mild winter of 2011, and above normal temperatures this spring, did not allow these insects to survive in Michigan, but much of the central United States has been above normal as well, giving some migratory insects a head start. For example, on March 22, the University of Kentucky reported armyworm moth catches in their pheromone traps at levels that are at least two weeks ahead of normal.

Insects that do overwinter in Michigan have ways to survive typical winter weather. Death by freezing isn't so much related to low temperature itself as it is the result of ice crystals forming in the body. Rapid formation and expansion of ice crystals cause cells to burst, resulting in organ and gut damage. Some insects are freeze-tolerant – they actually survive the formation of ice crystals in their body by producing ice nucleating proteins that "control" the freezing process.

Other insects are freeze avoidant – they accumulate antifreeze in their cells prior to the winter. The antifreeze is composed of specialized carbohydrates (in a fancy term, "cryoprotectants") that lower the freezing point of the body fluid, preventing the formation of ice crystals. Examples

of cryoprotectants are the sugars trehalose and mannitol, or the sugar alcohol glycerol (we humans use glycerol as an antifreeze in industrial processes). These cryoprotectants are effective as long as the insect body cools gradually (i.e., the insect acclimates to the cold, as in the fall, triggering the production of the compounds) and until temperatures get really cold (beyond the freezing point of the antifreeze).

To avoid exposure to severe cold and or fluctuating temperature, many insects overwinter under plant debris or burrow into the soil. As air temperature changes, the temperature under the cover rises and falls slowly (especially when insulated by snow cover), giving insects a far more stable environment.

Some examples: A first generation corn borer larvae collected in June is easily killed by cold. However, a second generation corn borer collected in December is freeze tolerant, and can survive for months at -4°F, even with ice crystals in its tissue. Overwintering eggs of many aphid species contain protectants like glycerol and mannitol to avoid freezing. In the case of soybean aphids, which spend the winter in the egg stage on exposed branches of buckthorn, eggs can be super-cooled to -29°F. Bean leaf beetles overwinter as adults, and typically survive temperatures only into the 20s°F. However, beetles overwinter in protected areas in woodlots or under leaf litter to avoid colder temperatures. In general, milder winter temperatures put less stress on these and other overwintering insects, and likely increase overall survival into the spring.

Once an insect successfully overwinters by avoiding freezing, it must successfully emerge, perhaps feed, colonize a crop, and eventually reproduce. A mild spring can help or hurt this process. For many adult insects (and some larvae) emerging from winter sleep, often the first task is to find food. Until food is available, they must live off of fat reserves stored in the body from the previous year. For other insects that overwinter as late-stage larvae, feeding is not an option; the fat reserves have to last through pupation, and even into the adult stage. If insects do not find food or complete development before energy reserves run out, the result is lower fitness, less reproduction, or even starvation. Thus being active too early or out of synch with a host crop can lead to reduced overall fitness. For example, alfalfa weevils emerging now in southern Michigan will likely find legumes to eat. But ladybird beetles that emerge early may not find enough prey to survive.

Early insect emergence often times coincides with earlier green-up of perennial crops or bud break on overwintering hosts, giving the insect population a head start and leading to larger pest populations. However, a cold snap can still kill spring vegetation and set the population back. For example, in 2007 a hard freeze damaged emerging leaves of buckthorn. This reduced the feeding sites for soybean aphids that had just emerged on these leaves, and 2007 ended up as a low aphid year in the state, although initial spring populations were high. Likewise, early pest emergence may coincide with earlier planting of the host crop (based on degree days), again leading to larger pest populations. However, a cold or wet period can suddenly set planting or emergence back, so that the insect life cycle and crop are out of synch. For example, in some

years with delayed planting, corn rootworm larvae emerged into bare field or corn borer moths did not find tall enough corn to produce a large first generation.

So the bottom line is to be observant as the spring progresses. Chances are that we will see a few unusually large insect populations, or some population peaks occurring earlier than expected. But, there could be weather events in April and early May that kill insects, or create synchrony problems between insect life cycles and crops. From the perspective of many insects, this is just another year in a bug's life.

For another extension article on this subject, see "Mild Winter, Record-Breaking March Temperatures: How Will Field Crop Insects Respond?" in the March 22 edition of The Bulletin from the University of Illinois.

Gluttons for punishment on this subject can read "Insect overwintering in a changing climate" from the Journal of Experimental Biology.

This article was published by **Michigan State University Extension**. For more information, visit http://www.msue.msu.edu. To have a digest of information delivered straight to your email inbox, visit http://bit.ly/MSUENews. To contact an expert in your area, visit http://expert.msue.msu.edu, or call 888-MSUE4MI (888-678-3464).

10 points:

Question: As we know, many anticancer treatments, such as gamma irradiation, or some chemotherapy agents (cisplatin etc) may by themselves cause cancer in healthy humans. What is the reason, and does it mean every anticancer therapy should be potentially carcinogenic?

Solution: Gamma-radiation or cisplatin damage cellular DNA. That is their major effect, and that is the reason why they are both anticancer and carcinogenic agents. To understand why they suppress the growth of cancer cells, we have to remember that cancer cell are dividing like crazy: they grow rapidly, then they divide, grow, divide again, and so on. To divide, any cell has to copy its DNA in full, otherwise newly formed cells will not be viable. Clearly, when chromosomal DNA is broken, its correct copying is impossible, and the cell's division will be aborted, and the cell will die. Our normal cells divide much slower than cancer cells, so normal cell have time to heal damaged DNA before the next division starts. Moreover, many cells in our organism, e.g. neurons are dividing so slowly that DNA breaks pose no danger for them at all. That is why chemical reagents (e.g. cisplatin) or physical factors (i.e. gamma-radiation) that cause DNA break have much greater negative effect on cancer cells than on our normal cells.
However, DNA breaks and DNA lesions are not absolutely harmless. Even in normal cells they may lead to mutations that give a start to a transformation of a normal cell to a cancer one (for

example, by activating so called *oncogene* proteins). However, this negative effect is much weaker than the effect of suppression of cancer cells.

COMPUTER SCIENCE

- You can write and compile your code here: http://www.tutorialspoint.com/codingground.htm
- Your program should be written in C, C++, Java, or Python
- Any input data specified in the problem should be supplied as user input, not hard-coded into the text of the program.
- Please make sure that the code compiles and runs on http://www.tutorialspoint.com/codingground.htm before submitting it.
- Submit the problem in a plain text file, such as .txt, .dat, etc. **No .pdf, .doc, .docx, etc!**

5 points: Neat Words.

Some words in the English language are just "neater" than others: their letters are tidily arranged in alphabetical order. Take *chinos* as an example. A good example of how things should be organized in your room, isn't it? Then again, some words are clearly rebellious: they arrange their letters in reverse to the alphabet, e.g. *yolked*.

Your task this month is to write a program that checks the words for neatness. You will receive a list of words on the input, and on the output you should indicate for each word whether its letters are neat, messy or rebellious. For example, given the following input:

```
sponged fiddle begins yellow yolked biopsy
```

you should produce the following output:

```
sponged IS A REBEL
fiddle IS MESSY
begins IS NEAT
yellow IS MESSY
yolked IS A REBEL
biopsy IS NEAT
```

Solution:

Python:

```
from __future__ import print_function
import sys
```

```
import re
while True: # we'll try forever until we get valid input
  # input words
  print("enter words separated by white space:")
  line = sys.stdin.readline()

  # verify that the input contains only words and white spaces
  matched = re.search("([^\\sA-Za-z])", line) # searching for NON white spaces or letters
  if matched:
       print("invalid input: '{}'; try again".format(matched.group(1)))
  else:
       break
pass
# split the input line into separate words
words = line.split()
for word in words: # process each word
  letters = list(word) # split word into letters

  # check if the word is neat
  is_neat = True
  for i in range(1, len(letters)):
       if letters[i-1] >= letters[i]: # nothing was said about the same letter; let's assume
strong neatness
       is_neat = False
       break
  if is_neat:
       print(word + " is neat")
       continue # next word
  # check if the word is rebellious
  is_rebellious = True
  for i in range(1, len(letters)):
       if letters[i-1] <= letters[i]:
       is_rebellious = False
       break
  if is_rebellious:
       print(word + " is rebellious")
       continue
  print(word + " is messy")
print("end.")
```

Java:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Words {
  private String[] words;

  public void readInput() throws IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    Pattern p = Pattern.compile("([^\\sA-Za-z])"); // searching for NON white spaces or letters
    for(;;) { // we'll try forever until we get valid input
      // read input line
      System.out.println("enter words separated by white space:");
```

```
      String line = br.readLine();

      // verify that the input contains only words and white spaces
      Matcher m = p.matcher(line);
      if(m.find())
        System.out.println("invalid input: '"+m.group(1)+"'; try again");
      else {
        words = line.split("\\s+");
        return;
      }
    }
  }

  public boolean isNeat(String word) {
    for(int i=1; i<word.length(); i++) {
      char prevLetter = word.charAt(i-1);
      char letter = word.charAt(i);
      int prev = Character.getNumericValue(prevLetter);
      int cur = Character.getNumericValue(letter);
      if(prev >= cur) // nothing was said about the same letters; let's assume strong neatness
        return false;
    }
    return true;
  }

  public boolean isRebellious(String word) {
    for(int i=1; i<word.length(); i++) {
      char prevLetter = word.charAt(i-1);
      char letter = word.charAt(i);
      int prev = Character.getNumericValue(prevLetter);
      int cur = Character.getNumericValue(letter);
      if(prev <= cur) // nothing was said about the same letters; let's assume strong
rebelliousness
        return false;
    }
    return true;
  }

  public void process() {
    for(String word : words) { // process each word
      if(isNeat(word)) {
        System.out.println(word + " is neat");
        continue; // next word
      }

      if(isRebellious(word)) {
        System.out.println(word + " is rebellious");
        continue; // next word
      }

      System.out.println(word + " is messy");
    }
  }

  public static void main(String[] args) throws IOException {
    Words words = new Words();
    words.readInput();
```

```
    words.process();
    System.out.println("end.");
  }
}
```

10 points: Garden Watering Optimizer

You have a rectangular garden where crops are arranged on a grid with integer coordinates. Normally there is enough usual precipitation to keep your garden healthy. but once a drought struck, and it is wreaking havoc to your garden. You lost a lot of crops, and you need to save as many of the remaining ones by adding a sprinkler. Unfortunately, you can add only one as water during drought times is rationed. The sprinkler rotates 360° and has a fixed watering radius (integer) creating a circular watering area. Your task is to find an optimal location where to place the sprinkler so that the maximum number of surviving crops fall within its watering radius. The sprinkler can only be placed strictly on a grid, and if there is already a surviving plant at the point, that plant is killed.

Your program will take 3 integers as an input: two dimensions of the garden (N rows, M columns) and the watering radius R of the sprinkler. Then you are given a map of the garden, which is N strings, each of length M and consisting of either dots (‘.’), representing no surviving crops, or x’s, representing growing plants.
Additional rules:

- all the plants and the sprinkler have integer 0-based coordinates
- the sprinkler covers the plant if the distance from the sprinkler is less than or equal to the sprinklers radius. For example, the distance from (2, 2) to (3, 4) is SQRT(5), which is more than 2, and therefore (3,4) would not be covered by a sprinkler with R=2 placed on (2,2)
- if you place the sprinkler on a point with a crop, you destroy the crop, so handle accordingly
- in the event you find two or more placements that yield identical scores, pick any one of them

Example. You are given the following input:

```
6 7 2
..x...x
.xx..x.
...x..x
.x.....
..x..xx
x..x.x.
```

For this garden, the ideal location of the sprinkler is (2,2), which would cover 6 plants.

As a bonus, output the map showing location of the sprinkler and watered plants, like this:

```
..X...x
.XX..x.
..OX..x
.X.....
..X..xx
x..x.x.
```

Solution:

Let's place the sprinkler in all possible places scanning the entire garden area. For each position we count how many plants it covers. Then we find the (first) maximum.
A naive implementation will place a sprinkler into a position and calculate spot by spot whether it is inside the circle or not using the circle equation:
x^2 + y^2 = r^2
The problem with this approach is that we need to perform a heavy calculation for each sprinkler position and for each possible plant spot. Although we cannot avoid scanning all sprinkler positions we can reduce the calculations for each spot or even examine less spots.
Our first optimization can be like this. We calculate a matrix for a covered area saving the result in a matrix with 0's (outside the circle) and 1's (inside the circle). This will be our pattern. We represent our garden as another matrix with 0's in empty spots and 1's in places where plants are. Then we'll move our pattern and stamp it over the garden area applying logical AND. If both elements are 1 meaning a plant is there AND it's inside the watering circle then we count it. If any of these two matrix elements are 0, i.e. it's either outside the circle or there's no living plant there then we don't count it; effectively we add 0 (the result of logical AND) to our count. Another way to imagine this is like if you are painting a cartoon character (running across the screen) over a fixed background (the logical operations will be different though).
Further, we can notice that when we shift the sprinkler position by 1 step, say to the right, the sprinkled area will shift right by 1 cell. Therefore, we don't need to reexamine the entire watered area (our pattern) but only the boundary. The saving effect will be more pronounced, the more sprinkler radius is. In order to simplify handling the boundary conditions (when the watering disk is partly in the garden area) let's introduce a margin area around the garden. The optimization described in this paragraph is overkill for a small garden and handling the extra margin area is not worth it. What is mentioned in the previous paragraph will be fast enough. However, if the garden is large this extra margin will be relatively small and the savings of not recalculating the entire disk will prevail. You can think about a rule when you want to switch from a simpler algorithm to this one.
In garden.py we implement this shifting logic (without the logic of switching to a different algorithm).

Python:

```
from __future__ import print_function
import math
```

```
import sys
import re

def read_input():
  while True: # we'll try forever until we get correct input
    try:
      print("Enter number of rows, columns and radius (integers separated by space):")
      line = sys.stdin.readline()
      matched = re.match("^\\s*(\\d+)\\s+(\\d+)\\s+(\\d+)\\s*$", line) # expecting 3 numbers
      if matched:
        n = int(matched.group(1))
        m = int(matched.group(2))
        r = int(matched.group(3))
      else:
        raise Exception("invalid input")

      # input garden map
      a = [['.' for j in range(m)] for i in range(n)]
      print("Enter your map with 'x' and '.' line by line:")
      for i in range(n):
        line = sys.stdin.readline().rstrip('\n')
        if len(line) > m:
          raise Exception("line must not exceed m")
        chs = list(line) # split line into characters
        j = 0
        for j in range(len(chs)):
          if chs[j] == '.':
            a[i][j] = 0 # dead spot
          elif chs[j] == 'x':
            a[i][j] = 1 # living spot
          else:
            raise Exception("line must contain only 'x' or '.'")

        for j in range(len(chs), m):
          a[i][j] = 0 # user didn't type white spaces at the end; defaulting to '.'

      return(n, m, r, a)
    except Exception as e:
      print("{}; try again".format(e))
  pass

def read_input_fixed():
  n = 6
  m = 7
  r = 2
  a = [[0,0,1,0,0,0,1],
       [0,1,1,0,0,1,0],
       [0,0,0,1,0,0,1],
       [0,1,0,0,0,0,0],
       [0,0,1,0,0,1,1],
       [1,0,0,1,0,1,0]]
  return(n, m, r, a)

# calculate boundary of a half circle of radius "r"
def calc_boundary(r):
  b = [math.floor(math.sqrt(r**2 - (i-r)**2)) for i in range(2*r+1)]
  return b
```

```
# scan the entire garden area "a" putting a sprinkler with radius "r" into all cells
# for efficiency use boundary indices "b"
# return matrix "counts" that for each sprinkler position contains number of watered plants
def scan_area(r, a1, b):
  assert(len(b) == 2*r+1)
  # we assume (0,0) is at the left top corner
  # our margin size is 2*r
  n = len(a1) + 4*r
  m = len(a1[0]) + 4*r + 1 # +1, so we always have previous
  counts = [[0 for j in range(m)] for i in range(n)] # initialize
  a = [[0 for j in range(m)] for i in range(n)] # new garden with margin
  # copy original garden: a[2*r:n-2*r][2*r+1:m-2*r] = a1[:][:]
  for i in range(len(a1)):
    for j in range(len(a1[0])):
      a[i+2*r][j+2*r+1] = a1[i][j]

  for i in range(r, n-r):          # scan row by row
    for j in range(r+1, m-r):      # column by column
      count = counts[i][j-1]       # previous value from left cell
      for k in range(len(b)):
        x = i - r + k              # add plants on the right semicircle
        y = j + b[k]
        count += a[x][y]
        #x = i - r + k             # subtract plants on the left semicircle
        y = j - b[k] - 1
        count -= a[x][y]
      count += a[i][j-1]           # restore killed plant by previous sprinkler
      count -= a[i][j]             # possibly kill plant by current sprinkler
      counts[i][j] = count
  # counts2 = counts[2*r:n-2*r][2*r+1:m-2*r] # remove margin
  counts2 = [[0 for j in range(len(a1[0]))] for i in range(len(a1))]
  for i in range(len(a1)):
    for j in range(len(a1[0])):
      counts2[i][j] = counts[i+2*r][j+2*r+1]
  return counts2

# draw garden "a" with the sprinkler at [x,y] with radius "r"
def draw_garden(a, x, y, r):
  print("garden with r = {}:".format(r))
  for i in range(len(a)):
    for j in range(len(a[0])):
      dist = math.sqrt((x-i)**2 + (y-j)**2) # distance between the current point and the sprinkler
position
      if abs(dist) < 1e-8:  # "dist==0" is incorrect; you cannot "exact" compare float numbers in
computer !!!
        print('O', end='')
      elif a[i][j] == 1:
        if dist <= r:
          print('X', end='')
        else:
          print('x', end='')
      else:
        print('.', end='')
    print()
  print()
```

```
#################

#n, m, r, a = read_input()
n, m, r, a = read_input_fixed()
b = calc_boundary(r)
counts = scan_area(r, a, b)
assert(len(counts)==n and len(counts[0])==m)

# find max index
maxx = max([max(x) for x in counts])
try:
  for i in range(n):
    for j in range(m):
      if counts[i][j] == maxx:
        raise Exception()
except:
  pass
print("max = {} at [{},{}]".format(maxx,i,j))

draw_garden(a, i, j, r)
print("end.")
```

Java:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class GardenMain {
  private int n;          // number of rows
  private int m;          // number of columns
  private int r;          // radius
  private int[][] a;      // garden map: 0 = dead spot, 1 = living plant
  private int[] b;        // semicircle boundary
  private int[][] counts; // for each sprinkler position contains number of watered plants
  private int maxx = 0;   // (1st) max number of living plants covered by sprinkler
  private int maxi = 0;   //   its index
  private int maxj = 0;   //   its index

  public int getMaxX() { return maxx; }
  public int getMaxI() { return maxi; }
  public int getMaxJ() { return maxj; }

  public void readInput() throws IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    Pattern p = Pattern.compile("^\\s*(\\d+)\\s+(\\d+)\\s+(\\d+)\\s*$"); // 3 integers (n, m, r)
    for(;;) { // we'll try forever until we get valid input
      // read input line
      System.out.println("enter n, m, r separated by white space:");
      String line = br.readLine();
      Matcher mt = p.matcher(line);
      if(!mt.matches())
        System.out.println("invalid input; try again");
      else {
        n = Integer.parseInt(mt.group(1));
```

```
      m = Integer.parseInt(mt.group(2));
      r = Integer.parseInt(mt.group(3));
      a = new int[n][m];

      System.out.println("enter garden map with 'x' and '.' line by line:");
      int i;
      loops:
      for(i=0; i<n; i++) {
        line = br.readLine();
        if(line.length() != m) {
          System.out.println("length must be "+m);
          break loops;
        }
        for(int j=0; j<m; j++) {
          char ch = line.charAt(j);
          if(ch == '.')
            a[i][j] = 0; // dead spot
          else if(ch == 'x')
            a[i][j] = 1; // living spot
          else {
            System.out.println("invalid character "+ch);
            break loops;
          }
        }
      }
      if(i == n)
        return; // got valid input
    }
  }
}

public void readInputFixed() {
  n = 6;
  m = 7;
  r = 2;
  a = new int[][] {{0,0,1,0,0,0,1},
                   {0,1,1,0,0,1,0},
                   {0,0,0,1,0,0,1},
                   {0,1,0,0,0,0,0},
                   {0,0,1,0,0,1,1},
                   {1,0,0,1,0,1,0}};
}

// calculate boundary of a half circle of radius "r"
public void calcBoundary() {
  b = new int[2*r+1];
  for(int i=0; i<2*r+1; i++)
    b[i] = (int)Math.floor(Math.sqrt(r*r - (i-r)*(i-r)));
}

// scan the entire garden area "a" putting a sprinkler with radius "r" into all cells
// for efficiency use boundary indices "b"
// calculates matrix "counts" that for each sprinkler position contains number of watered plants
public void scanArea() {
  assert(b.length == 2*r+1);
  // we assume (0,0) is at the left top corner
  // our margin size is 2*r
```

```
    int n2 = n + 4*r;
    int m2 = m + 4*r + 1; // +1, so we always have previous
    int[][] counts2 = new int[n2][m2]; // initialized to 0 by default
    int[][] a2 = new int[n2][m2]; // new garden with margin
    // copy original garden: a[2*r:n-2*r][2*r+1:m-2*r] = a1[:][:]
    for(int i=0; i<n; i++) {
      for(int j=0; j<m; j++) {
        a2[i+2*r][j+2*r+1] = a[i][j];
      }
    }

    for(int i=r; i<n2-r; i++) {      // scan row by row
      for(int j=r+1; j<m2-r; j++) {  // column by column
        int count = counts2[i][j-1]; // previous value from left cell
        for(int k=0; k<b.length; k++) {
          int x = i - r + k;         // add plants on the right semicircle
          int y = j + b[k];
          count += a2[x][y];
          y = j - b[k] - 1;
          count -= a2[x][y];
        }
        count += a2[i][j-1];         // restore killed plant by previous sprinkler
        count -= a2[i][j];           // possibly kill plant by current sprinkler
        counts2[i][j] = count;
      }
    }
    // remove margin: counts2 = counts[2*r:n-2*r][2*r+1:m-2*r]
    counts = new int[n][m];
    for(int i=0; i<n; i++) {
      for(int j=0; j<m; j++) {
        counts[i][j] = counts2[i+2*r][j+2*r+1];
      }
    }
  }

  public void findMaxIndex() {
    for(int i=0; i<n; i++) {
      for(int j=0; j<m; j++) {
        if(counts[i][j] > maxx) {
          maxx = counts[i][j];
          maxi = i;
          maxj = j;
        }
      }
    }
  }

  public void drawGarden() {
    System.out.printf("garden with r = %d\n", r);
    for(int i=0; i<n; i++) {
      for(int j=0; j<m; j++) {
        double dist = Math.sqrt((maxi-i)*(maxi-i) + (maxj-j)*(maxj-j)); // distance between the
current point and the sprinkler position
        if(Math.abs(dist) < 1e-8) // "dist==0" is incorrect; you cannot "exact" compare float
numbers in computer !!!
          System.out.print('O');
        else if(a[i][j] == 1) {
```

```
          if(dist <= r)
            System.out.print('X');
          else
            System.out.print('x');
        }
        else
          System.out.print('.');
      }
      System.out.println();
    }
    System.out.println();
  }

  public static void main(String[] args) throws IOException {
    GardenMain garden = new GardenMain();
    //garden.readInput();
    garden.readInputFixed();
    garden.calcBoundary();
    garden.scanArea();
    garden.findMaxIndex();
    System.out.printf("max = %d at [%d,%d]\n", garden.getMaxX(), garden.getMaxI(),
garden.getMaxJ());
    garden.drawGarden();
  }
}
```