***1) How many pennies could you put on the Golden Gate Bridge without any of them overlapping? For this question, please write out each step in your thought process.***

**Information Gathering**

The Golden Gate Bridge, Highway and Transportation District website says that the length of the suspension span including main span and side spans is 1 966 m and that the width of the bridge is 27 m (<http://www.goldengatebridge.org/research/factsGGBDesign.php>).

The US mint website says that a U.S. penny is 19.05mm in diameter (<http://www.usmint.gov/mint_programs/circulatingCoins/?action=CircPenny>).

Thus, we know that:

*L* = 1 966 000 mm

*W* = 27 000 mm

*dpenny* = 19.05 mm

Gauss has shown that a hexagonal lattice is the densest possible packing of circles on a plane. Therefore, from this point on, I will assume that pennies will be set out on the bridge in a hexagonal lattice pattern and, for ease of illustration and calculations, consider the pennies as hexagons of diameter *dpenny* = 19.05 mm.

**Overview**

Generally, my thinking was that I would start laying out a single row of end to end pennies lengthwise and then see how many additional rows I could add widthwise next to this initial row. Given that I am working with a hexagonal lattice pattern, the width of pennies gained for each additional row is unknown and will need to be calculated.

**Specifics**

The number of pennies that can be fit lengthwise can be calculated by:

*L / dpenny* = 1 966 000 / 19.05 = 103 202.1

Therefore, 103 202 pennies can be fit lengthwise with 1.9 mm of unused space remaining at one the end of the bridge.

If a second row composed of the same number of pennies (103 202) is added next to the first row, I notice that one penny on the end will overhang. Knowing that more than half of the penny ((19.05mm / 2) + 1.9mm) is resting on the bridge, I can assume that the penny will stay safely in place and so is still “technically” on the bridge. As I am trying to fit as many pennies on the bridge as possible, it seems in my interest to retain as many as possible, even if this means having some pennies overhang a little.

Pushing this reasoning further, if the first row were moved to be in the center of the bridge lengthwise, there would be 0.95mm (1.9mm / 2) of unused space on either end of the bridge. Then, adding the second row, I could add an additional penny, thus allowing every other row to contain 103 203 pennies, while still allowing enough space on the bridge so that at least half of all pennies are on the bridge. Specifically, the pennies at each end of the second row would have 10.475mm of their total 19.05mm diameter (54.9%) on the bridge.

In order to determine how many of these rows of pennies I can fit in the width of the bridge, I need to be able to calculate the width that is added each time I add a new row.

I start by looking at hexagon geometry and calculating the length of the hexagon’s sides. I know that:

*dpenny* = *t*(3)1/2

19.05 = *t* (3)1/2

19.05 / (3)1/2 = *t* = 10.99

Now that I know the length of *t*, I simply need to calculate the remaining added width (see left hand figure).

This can be accomplished by looking at the small section of the hexagon as a right angle triangle where I am looking for the height (*h*) of the triangle.

As I know the length of one side of the triangle (½ *dpenny*) and the hypotenuse (*t*), I can calculate the height using the Pythagorean Theorem, a2 + b2 = c2.

½ *dpenny*2 + *h*2 = *t*2

9.5252 + *h*2 = 10.992

*h*2 = 10.992 – 9.5252

*h* = 5.499 mm

Therefore, for each new row added, the width of pennies should increase by *t* + *h* = 10.99 mm + 5.49 mm = 16.49 mm.

Thus, *W* = *dpenny* + *x*(*h* + *t*)

where *x* is the number of additional rows that can be added to the first row of pennies to span the width.

For *W* = 27 000 mm, *x* = 1635.429

A total of *x* + 1 = 1636 rows (*x* + the first row) can comfortably be fitted onto the bridge. However, as with row length, the question of possible overhanging pennies should be addressed.

For 1636 rows of pennies, the width of bridge occupied is equal to the width of 1st row + 1635 \* width of additional rows:

19.05 mm + 1635\*(16.49 mm) = 26 992.93 mm

Thus there remains 7.07 mm (27 000 mm - 26 992.93 mm) of unoccupied space on the bridge at the end of the rows.

If another row (row 1367) is added, 12.57 mm (7.07 mm + *h* = 7.07 mm + 5.49 mm) is the width of the penny that is still on the bridge. As this is greater than half of the penny’s diameter (65.99%), we can accept this additional row.

The question remains if all rows were moved to the left so that the first row would overhang, would it be possible to add yet another row on the right. This would imply moving all rows by <½ *dpenny* or <9.525 mm to the left which translates to the unoccupied width on the right (previous to the 1637th row being added) being increased to <(7.07 mm + 9.525 mm) or < 16.59 mm.

As the addition of the 1637th row will take up 16.49 mm of the unused space, this leaves <0.1 mm (16.59 mm – 16.49 mm) of unoccupied space at the edge, thus leaving only *h* + <0.1mm or <5.59mm of space for the pennies of an additional row (1638th) which is less than half of the penny’s diameter (<29.39%), thus not enough for it to remain safely on the bridge.

**Conclusion**

And so, in the end, we will have 1637 rows, 818 of which will be of composed of 103 202 pennies and 819 of which will be composed of 103 203 pennies (unlike the illustrations, the 1st row will have 103 203 and the second will have 103 202).

818 \* 103 202 + 819 \* 103203 = **168 942 493** pennies

Out of curiosity, dividing the area of the bridge by the area of a hexagon of d = 19.05 gives 168 898 847.43 which is close but less than my result. I expect that the overhanging pennies explains this.

Furthermore, out of more curiosity, I verified if making the rows widthwise instead of lengthwise might lead to a greater number of pennies. In this scenario, 168 917 805 pennies which is less than that calculated with widthwise rows. This result is not unexpected as I estimate the amount of pennies gained by adding an additional row is greater than the pennies gained by adding a penny to a row, thus it makes sense that this be taken advantage of by increasing rows on the longer dimension of the bridge, namely the length.

***2) Our images have a ratio of 16:9, and our design layouts have 12 pixel wide increments (there are no limits on height). Give examples of three image sizes that would have the correct ratio and would fit the design layout.***

If pixel wide increments are 12px, then all image widths must be divisible by 12 and 16. The least common denominator for 12 and 16 is 48, which represents the minimal possible image width, in pixels, for the design layout. Given the 16:9 ratio, the smallest image possible would be 48px wide by 27px high. All other images would be multiples of this image size:

96 x 54, 144 x 81, 192 x 108, 384 x 216, 768 x 432, 1536 x 864, etc.

***3) What is the minimum number of moves required for a knight to cover the entire chess board? Write a program to prove it. Provide a graph with the move number on the x axis and the number of squares covered on the y axis.***

64 moves is the minimum. The program can be found in the Chess1 directory on my GitHub repository IGN-code-foo.

I started from the assumption that the least hypothetical amount of moves would be equal to the amount of squares on the board. In this case, 8 x 8 = 64 squares, and so 64 moves. Specifically, this could only happen if the knight were able to cover all squares without ever playing a square more than once. If when considering the knight’s L-shaped move pattern, this were not possible, then the next least amount of moves would imply playing one single square twice and all other squares only once. And so on, until the entire board could be covered.

*Therefore, the first critical variable to take into account is the number of times the knight has played squares more than once (MaxMultMoves in the program).*

As the starting position might influence whether or not it would be possible for the knight to cover the entire board for a given number of times that squares are played more than once, every possible starting position was tried out before increasing the number of allowable squares played more than once.

*Therefore, the starting position is the second critical variable to be taken into account (StartPos in the program)*

The program I created basically starts in a given position, evaluates all possible moves from this position and selects the first possible move by default. *MaxMultMoves* controls how often a square can be replayed (0 – 999). If it comes to a situation where there are no possible moves, the knight retreats by 1 move, then tries the next possible move. This continues until either the knight is back at its starting position or all squares on the board have been played at least once. In the former case, the loop is broken and the starting position changed. If all starting positions have been tried, the number of squares that can be replayed is increase by 1. In the latter case, the program ends and the following information is displayed:

* The number of allowable replayed squares (*MaxMultMoves*)
* The current move number
* A board with the number of times a square has been played
* A board with the move numbers on each square
* A list of all positions played

After running the program, I can say that starting at position (0,0), it is possible to cover all squares in 64 moves.

The following graph shows the number of squares covered in relation to the move number (Move number 0-63)

***4) Creatively prove to us that you meet our value – Fire – that this would be more than just a job to you, and that you are passionate about us. Why do you need IGN and why does IGN need you?***

A little about myself: I’ll save you the “I have been interested in gaming since I was a child” routine as I’m sure 99% of the applications you’re reading start that way. I started programming on my Amiga 500 when I was in elementary school, mostly making small Basic programs that would do my math homework for me or making simple games. I went to college and studied electronics and learned some Assembly language programming. The program was a mix of electronics and computer science. One term project involved building a simple computer from logic circuits and other electronic components and then programming it to perform simple actions like moving a robotic arm.

I stayed in the program for two and a half years, but I switched to biology in university where I completed my undergraduate and master’s degrees. For my graduate work, I used georeferenced data to build large-scale prediction models of caterpillar behavior. I know it sounds a little esoteric and unrelated, but a lot of my work involved working with statistics programs, GIS software, etc. all of which required some level of tweaking through programming in order to have it do exactly what I needed.

Since then, I’ve been working in my field doing contract work with different organizations. During one of these contracts, my employer asked me if I knew anything about web design as she wanted to redo her website. I told her that I didn’t know very much but if she gave me a chance, I’d figure it out and put something together. I went out and bought some books, did some online research and one month later, I had managed to teach myself HTML and CSS and was beginning to get a good grasp of Java. I was also gaining proficiency using Photoshop and creating custom content (backgrounds, buttons, etc.). The site I created for her research project can be found at <http://web2.uqat.ca/safe/> although all the content is in French. For a first attempt at a website, I think it looks pretty good. Over the next couple of years, I took occasional web design contracts and continued to develop my skills.

Currently, I am planning on beginning a Ph.D. program at the University of Arizona in September. My thesis topic will be on social insects, specifically ants, and building agent-based models looking at individual behavior to explain colony-wide behavior. I am also hoping that I can use these agent-based models in an AI programming framework. As such, I am looking at including a significant portion of programming. My secret hope is that this work leads to applications in the gaming industry, developing AI for video games.

One thing that has always struck me about IGN is how it seems to think outside the box. In particular, I’m thinking about how you seem to go out of your way not to necessarily hire the people who look best on paper, but people who are passionate about their work because you know that they’ll put the effort in to do a good job. The story of how IGN recruited Jessica Chobot is a well known example of this, but I was reading an article recently about how different people had started working at IGN. I can’t seem to find the article now, but I remember this one particular story about a kid who, when he was in high school spent this ridiculous amount of time writing a game guides and was eventually recruited by IGN for an internship and one thing led to another and was now an editor at IGN. That’s a great story about a person who is passionate about something and a company being able to recognize the value of that.

I am definitely passionate about the gaming industry. I get excited each time I find a new little snippet of information about whatever video game I’m currently obsessed with. At the moment, it’s Skyrim and I think the next six months are going to be some of the longest months of my life. Every time a new game I’ve been ever-so-anxiously waiting for comes out, I send a general disclaimer to family and friends informing them that I’ll be dropping off the face of the earth for the next little while and go on a gaming binge, dedicating every spare minute I have to play “just one more turn” or solving the next case and getting the promotion to the vice department. And yes, I just came off an L.A. Noire binge to work on these questions. I guess what I’m saying is that I think I’d fit right in…

I also like how IGN is willing to give people a chance to prove themselves. This internship is a perfect example of this. I really appreciate that the internship is open to anyone. I also appreciate that you’ll be making decisions based on how people answer these questions. These are great questions and I must say that I really enjoyed answering them. I wish I had more time to spend on the pennies on the Golden Gate Bridge problem as I see a lot of potential there. I was reading a couple of articles on circle packing theorem and different heuristic and exact methods that exist for solving these types of problems. Unfortunately, there wasn’t enough time to thoroughly understand and apply these methods. Hopefully my geometric approach provides an acceptable optimization of space.

I only found out about this internship a little over a week ago, so I’ve been working pretty hard these last few days to put together these answers. I started with the problem of the knight covering all the pieces on the chess board in the least moves possible. I began by writing a code in R (a programming language for statistical analyses based on S) which is probably the language that I’m currently most comfortable with. I got it working alright, but after a little bit of testing, I realized that running it for 100 million iterations would take something like 35 hours of processing time and wouldn’t even make a dent in all the possible move combinations.

So, I decided to bite the bullet and learn some C++. Over this last weekend, I scoured the internet for C++ beginner’s guides and tutorials and I managed to transpose my program to C++. I anxiously started it up and let it run, hoping that it would come up with an answer before the application deadline. After a grand total of 8 minutes processing, I had my answer. Now, I know my code is probably not perfect, but what I’m getting at is that, if given a chance, I can take on anything that you throw at me. That’s the way I am. Give me a problem and sooner or later, I’ll have a solution. I am motivated when challenged and am pigheaded enough to do what it takes to get it done, even if that means teaching myself C++ in a weekend.

So, why do I need IGN and why does IGN need me? You need me because I am truly passionate about the work you do and would definitely be an asset. I work hard and I thrive on challenge. Mostly, you need me because I can get it done and done well.

I need IGN because I’m passionate about working in the gaming industry and have the drive to do what it takes to make it happen. I feel like this internship is perfect for me because it is an opportunity for me to develop my skills in what I expect is exactly the right environment. I think that IGN would provide a work environment where my desire to learn and improve would be fostered, where excellence is rewarded and being yourself, being original is encouraged.

I realize that I don’t have a typical programming background and it might not be an obvious match, but it seems to me that the IGN team is composed of a “pack of strays”, akin to the Belafonte crew. Do all interns get Glocks, I wonder?