AP Computer Science Principles
Summer 2017 Assignment

Welcome to Computer Science Principles (CSP)!!! This class is about the principles that underlie much of the technology around us. Your job as a student in the class is to be on the lookout, to be alert, for where and how computer science affects or impacts the things you care about, the things you know about. Whatever it is you want to do in your life, it's likely that technology affects it in some way, or that some technological innovation is needed.

This course is about creativity, technology and innovation. In this class, you will often be asked to invent your own solution to problems. Even if it's a problem that has been solved before, thinking like a computer scientist requires a different mindset. We study a lot of things in this class, not only because its foundational knowledge, but because of the way it makes you think, the way it asks you solve problems.

Inventing things, and having insights about how things work, and how they might work better is what this class is about. The goal of the summer assignment is to get you started on this way of thinking. It consists of a reading assignment along with some questions (don't worry, it's not that long) and producing an artifact to turn in at the beginning of class.

Directions:

1. Obtain a copy of the book *D is Digital* by Brian Kernighan. You may purchase from Amazon or if you have Kindle Unlimited, you can get it free.
2. Read the first 5 chapters of this book. That's all I want you to read: 5 chapters; stop at page 83 (unless you can't put the book down, in which case, feel free to read as much as you like). This is about the first 1/3 of the book. We may use the rest of the book throughout the school year. Then complete the reading guide. This will not be turned in, but we will have an assessment of the material the first week of class.
3. Select a computer science innovation **introduced before 2007** that had an impact on your current way of life and research the positive and negative impacts it had on our world. Here is a sampling of sites you might use to help you choose a computer science innovation:
    - Timeline of Computer History
    - https://compsci.lafayette.edu/homepage/top-30-innovations/
    - http://www.nytimes.com/topic/subject/science-and-technology
    - http://www.livescience.com/technology
    - http://news.mit.edu/topic/computers
    - http://www.brookings.edu/blogs/techtank/posts/2014/06/10-mit-techreview-2014
    - https://www.sciencedaily.com/news/computers_math/computer_science
4. **Use your creativity!** Produce a computational artifact (a visualization, graphic, video, or audio recording) that illustrates, represents or explains the computing innovation's intended purpose, its function, and its effect on our lives. Try to find a creative way of conveying the results of your research. You might create an infomercial or public service announcement or a cartoon/comic strip.
   This portion of your summer assignment is a warm-up for one of the required Performance Tasks that must be completed as part of the AP exam. The more effort you put into this summer assignment, the easier it will be to complete the AP Performance Task. The artifact you create should adhere to the same AP requirements as the Performance Tasks. Acceptable multimedia file types include .mp3, .mp4, .wmv, .avi, .mov, .wav, .aif, or .pdf format. PDFs must not exceed 3 pages in length. Video or audio files must not exceed 1 minute in length and must not exceed 30MB in size.
   Below are some tools you may consider using to create your computation artifact (you are not limited to these)
    - Use Adobe Spark to create an animated presentation

- Create a presentation using Google slides or any slide tool and then use one of these Screencasting tools to record an audio/visual artifact
- Use WeVideo to create a multimedia presentation

If you create a video, most of these tools will create a .webm video file, but it needs to be an .mp4 file. Use this online converting tool to convert your video to .mp4 format.

YOU WILL RECEIVE A GRADE BASED ON THE RUBRIC ON THE NEXT PAGE

5. Create a MS Word or Google doc with the following:
   - Describe your development process, explicitly identifying the computing tools and techniques you used to create your artifact. Which tool(s) did you use? Why did you choose it? What was good about the tool(s) you used? What was difficult to do with the tool(s) you chose? Would you recommend these tools to others? Why or why not?
   - References: Provide a list of at least three online or print sources used to support the information in your computational artifact
     - Include citations for the sources you used
     - Each source must be relevant, credible, and easily accessed
     - At least 2 of the sources must have been created within **2 years before or after the innovation was introduced**
     - For each online source, include the permanent, URL. Identify the author, title, source, the date you retrieved source, and if possible, the date the reference was written or posted
     - For each print source, include the author, title of excerpt/article and magazine or book, page number(s), publisher, and date of publication
   - If using a Mac, and using Pages to create this document, convert the .pages file into a MS Word .doc(x) file before uploading. If you need help with this please contact Mrs. Beaman.
6. Upload both your computation artifact and your Google doc by the first day of school. (location to be determined by the beginning of August)

If you have any questions, contact Mrs. Beaman at cbeaman@parrottacademy.org, or come out to the school. I will be around most of the summer!

Summer Project Rubric

Criteria for the Computational Artifact	**Points**
What is the technological innovation? The artifact specifically identifies the innovation and explains the role computing plays in the innovation.	/4
What is the most important thing that it enables you to do? The artifact precisely describes the personal significant impacts experienced.	/4
How did people do this before the technology existed? The artifact fully identifies how people used to perform the identified task.	/4
What was a problem or issue with the way it was done before that this technology solved or made better? The artifact precisely describes significant impacts felt before the innovation became available and identifies how those impacts have been eliminated or reduced by the innovation.	/4
What new problems were created by the technology? The artifact presents a rich analysis of problems created by the innovation.	/4
Describe your development process, explicitly identifying the computing tools and techniques you used to create your artifact. ***Acceptable multimedia file types include .mp3, .mp4, .wmv, .avi, .mov, .wav, .aif, or .pdf format. PDFs must not exceed 3 pages in length. Video or audio files must not exceed 1 minute in length and must not exceed 30MB in size.*** Which tool(s) did you use? Why did you choose it? What was good about the tool(s) you used? What was difficult to do with the tool(s) you chose? Would you recommend these tools to others? Why or why not?.	/4
Provide a list of at least three online or print sources used to support the information in your computational artifact ● Include citations for the sources you used. ● Each source must be relevant, credible, and easily accessed. ● At least two sources must have been created within two years before or after the innovation was introduced ● For each online source, include the permanent URL. Identify the author, title, source, the date you retrieved the source, and, if possible, the date the reference was written or posted. ● For each print source, include the author, title of excerpt/article and magazine or book, page number(s), publisher, and date of publication	/6
Total:	/30

This book is an introduction to the areas of computer science that we will cover in greater detail during the school year. I don't expect you to remember or understand everything the first time through. That said, you should find this material interesting and engaging. If you think it's amazing that *everything* we see and do on a computer comes down to sequences of 0s and 1s processed millions of times per second, then you will enjoy this read and the class.

This book is the perfect way to begin looking under the hood at a computer and understanding how the technology that we so often take for granted actually works. I hope you truly enjoy this read. As you progress through the chapters, keep in mind the questions below. Mark their answers, and jot them down in the margins, using the back of the pages or other paper as needed. **Think of this as an annotation guide; answers do not need to be in full sentences.**

Preface

- What are the three core technical areas of computing? (Hint: they form the three parts of this book.)
- What is a potential fourth, according to Kernighan?

Introduction

- What is one of the many significant technical innovations of the 2000s that Kernighan explains? How did it change and/or improve upon previous technology?
- Kernighan identifies three important underlying ideas to how computers and communications systems work. What are they?

Part I: Hardware

- Who is often referred to as the world's first programmer? What is she known for?

What's in a Computer?

- What is the role of the CPU?
- If you have a 1.7 GHz computer, what does the "1.7 GHz" part mean?
- What does the "mega-" mean? What does "giga-" mean?
- What is the role of RAM? What information does it store?
- What is the difference (or trade-off) between storing something on a disk versus storing something in RAM?
- The most fundamental element of a computers electronic circuitry is a logic gate. What is a logic gate? (Check out the video for logic gates in the Crash Course Computer Science series on YouTube.)
- What is Moore's Law?

Bits, Bytes, and Representation of Information

- What are the three fundamental ideas about how computer represent information?
- What is the difference between “analog” and “digital”?
- What is a pixel?
- The iPhone 7 has a 12-megapixel camera and a screen with 1334-by-750-pixel resolution at 326 ppi (pixels per inch). Based on your understanding of a pixel, what do these tech specs mean in simple terms?
- Chances are that you have at least one digital file of music on your computer. See if you can find out it’s bit rate and sample rate. What are they? (In iTunes, go to **Edit** > **Get Info** > **File**)[1]
- What is ASCII? What is Unicode?
- Unicode makes emojis possible. See if you can find the most recently created emojis. What are their Unicode values? (Bonus: how many bits/bytes does it take to represent one Unicode character? How does this contrast to ASCII?)
- What is a “bit”?
- Why does a power switch have a 0 and a 1 on it? What state does each represent?
- If you have *N* bits, how many different patterns of 0s and 1s can you represent?[2]
- Let’s say you have a 1-terabyte hard drive. This could mean two different things (controversially). What are they two possible values for 1-terabyte based on powers of two and powers of ten, respectively?
- What is a “byte”? How many different values can be encoded in 1 byte?
- What is the hexadecimal number system?[3]
- Where is the most common place to come across hexadecimal?
- Kernighan notes a “critical thing” near the end of section 2.3. Mark this paragraph. It is an idea fundamental to this course. What is his point here?

[1] This information is called metadata, a key term for our course, especially when it comes to digital media.
[2] At this point the material might start to seem a little more complex. That’s okay for now. Absorb what you can. We will spend a lot of time going over the binary system in class.
[3] This information can be even more complex. Again, understand what you can on your own. We will spend time going over it in class together.

Inside the CPU

- What is the function of an accumulator?
- Kernighan's explanation of the "Toy" computer will be a helpful example when it comes time for us to write and understand code in the C programming language. There are also helpful computational thinking ideas in this section. For example, what is the purpose of "`GOTO`"? "`IFZERO`"?
- The bottom of page 39 has some essential ideas for how to construct programs. What key ideas emerge here?[4]
- In very simple terms, what is the fetch-decode-execute cycle?
- In the context of computer memory, what purpose do caches serve?
- Kernighan gives you a fun Google trick to try at the end of 3.3. Come up with an obscure, seemingly random phrase, and note the time it takes Google to respond. You'll see this at the top of the results. Then, run it again immediately. What is the time differential? Caching at work!![5]
- Who was Alan Turing? What is the "Turing test"?
- What does CAPTCHA stand for?

Part II: Software

- What is software? How is it different from hardware?

Algorithms

- How does Kernighan define "algorithm"?
- Kernighan gives a brief aside to data structures. What is his short explanation for this term?
- What does the term "linear-time" have to do with algorithms?
- What is binary search? What real world application does it have?
- Sorting algorithms will be a focus of our study later in the first semester. For now, which algorithm is better? In simple terms, show does this better algorithm work? That is, what is its general approach to sorting?
- What is the Traveling Salesman Problem?
- Look up the word "heuristic" in the context of computer science. How does it relate to the end of 4.4?

[4] Notice how you don't need to program to get better at the logic of programming; they key to being a good programmer is not knowledge of syntax, but logical and ordered thinking.
[5] I Googled the phrase "hippopotamus homeric coffee." The first search took 1.08 seconds, the second only 0.63. See how high you can get the response time. Anything > 1 second is an accomplishment.

Programming and Programming Languages

- What is the difference between an algorithm and a program?
- What is the function of and assembler and assembly language?
- Are assembly language instructions the same for all computer processors?
- What is the function of a compiler?
- What are the advantages of a high-level languages relative to assembly language?
- What are the five high-level languages Kernighan demonstrates code with? Which seems hardest to read and understand? Easiest?[6]
- What is a library? What is a library's API?
- What is the term for a flaw in one's code? What is the origin of this term as it relates to computer?
- What are general definitions for the following terms:
    - Trade secrets
    - Copyright
    - Patents
    - Licenses
- What is a "standard" in the world of computing?
- What is the relationship between source code and object code?
- What does it mean for code to be "open source"?
- What is one example of open source software?

Try to read over the material more than once over the summer to let it really sink in. Explore some tech specs of your own computer(s). If there is more than one in your house, compare the differences between them. Read about the history of Linux (which our CS50 IDE[7] uses as its operating systems). These 5 chapters will prepare us to jump right into Unit 0 on Day 0 in August.[8]

[6] For your reference, CS50 uses C and Python as its primary languages. Other than a quick introduction to programming using Scratch, all of first semester will be spent focused on learning C.
[7] IDE stands for Integrated Development Environment. In simple terms, it is where we write and execute our programs.
[8] Computer scientists start counting at 0, not 1.