COMPUTER SCIENCE 110

**INTRODUCTION TO COMPUTER SCIENCE**

**HOMEWORK 1 (Multimedia Python)**

# Spring 2018

**Assigned**: Monday, 1/22 **Due**: **by beginning of class** Monday, 2/5

**Answer the homework questions on a separate page. You do not need to turn in this sheet**. **For the programming portion, follow the directions in that section.**



**Part I (40 points) Introduction, Number Bases** (**show your work for full credit; it’s all right to hand-write the work, but be neat (or type if you can’t be neat): take pride in the product of your efforts!). Your answers must be on a separate sheet of paper; do not turn in your answers on this sheet**:

1. CSI textbook, Chapter 1, Thought Question 1 (p. 39). Supply only 2 examples, not 5. (8 pts)

Example of three abstractions in school environment (only need 2):

* Blackboard gradebook – you see a nice layout showing your grades for a particular class. In the background, there is a lot of work going on that links your grades to a particular class and a particular instructor. The instructor enters much more information and the computer takes it and gives it to you in a readable format. You don’t have to learn how to query a database to get to this information.
* Cafeteria: you see prepared food and have a place to dispose of used dishes, etc. Details of the cooking are hidden (you don’t have to cook or worry about how it’s done, and you’re not getting in the way of the cooks). Details of turning dirty dishes into clean ones are also hidden behind a wall – you don’t have to wash them or even think about how they get washed.
* Buildings – details of how they are heated, cooled, cleaned, and maintained are largely hidden from you. This makes your life simpler so you can focus on what’s important (your coursework!)
1. Convert the following **sign-magnitude** numbers from binary to decimal (base 10). Assume that a “1” in the leftmost bit indicates a negative sign (6 pts)
	* 1. 101011101 -93
		2. 011111011111 +2015
		3. 1101001001 -329
2. Fill in the following chart with equivalent values in each cell. For example, for the first line, convert the numbers from decimal to binary and then to hexadecimal and octal. **Do not use sign-magnitude notation.** (12 pts):

|  |  |  |  |
| --- | --- | --- | --- |
| Base 10 | Base 2 | Base 16 | Base 8 |
| **99** | 1100011 | 63 | 143 |
| 33 | **100001** | 21 | 41 |
| 3499 | 110110101011 | **DAB** | 6653 |
| 239 | 11101111 | EF | **357** |

1. What is the range of **signed** decimal values that can be represented using
	* 1. 4 bits? 15 values can be represented, -7 to +7
		2. 2 bits? 3 values can be represented, -1 to +1
		3. 2 bytes (not bits!)? 65535 values can be represented, -32767 to +32767

 Express your answers by both stating how many values can be represented and showing the lowest and the highest, in **base 10. Please do *not* express your answer using exponentiation** (6 pts).

1. CSI Textbook, Chapter 2, Exercises 35 and 36 (p. 57) (8 pts)

Answers may vary:

35. Could use the following substitutions for the numbers 10-17:

 10 !

 11 @

 12 #

 13 $

 14 %

 15 ^

 16 &

 17 \*

36. a) 1066 would be 354

 b) 99099 would be &\*^9

 c) 1 would be 1



**Part II. (25 points) The Last Lecture**

On September 18, 2007, Carnegie Mellon professor and alumnus Randy Pausch, the inventor/driving force/inspiration behind the Alice programming language, delivered a one-of-a-kind last lecture that made the world stop and pay attention. It became an Internet sensation viewed by millions, an international media story, and a best-selling book that has been published in 35 languages. To this day, people everywhere continue to talk about Randy, share his message and put his life lessons into action in their own lives. He passed away in July 2008 of pancreatic cancer.

An excerpt from the talk:

*“Almost all of us have childhood dreams: for example, being an astronaut, or making movies or video games for a living. ...  Sadly, most people don't achieve theirs, and I think that's a shame. I had several specific childhood dreams, and I've actually achieved most of them. More importantly, I have found ways, …, of helping many young people actually \*achieve\* their childhood dreams. This talk will discuss how I achieved my childhood dreams … and will contain realistic advice on how \*you\* can live your life so that you can make your childhood dreams come true, too. “*

 So, what are the guidelines for this question?

1. To access the video, go to <http://www.cmu.edu/randyslecture/> and watch the video.
2. Turn in **at least** 1 1/2 pages (typed, 12 pt., double spaced, times new roman or arial font, no larger than 1” margins) discussion of what parts of the video stood out the most.
3. Discuss somewhere in the response the advice his advisor gave to him about dealing with other people.  How did his advisor present this piece of advice and what was another way (less tactful) that he could have said the same thing?
4. What had the most emotional impact on you, personally?  How did this speak to you?
5. Who was the real audience for his lecture?



**Part III (35 points) Python Programming**

Your Python code must follow these guidelines:

* *Always put your name, date, program name, and a general comment describing what your program does at the top of your code. This should be a general statement about how your program behaves.*
* *Variable names should help describe the purpose of a variable.*
* *Function names should help describe the purpose of a function*
* *Every function should be preceded by a comment describing the function’s purpose, input, and output (if any). This is true whether you wrote the function yourself or not.*

Be aware that if you don’t follow these guidelines, you will lose points, **even if your program runs perfectly**.

Write a program named **createSequence()** to create a sequence of the same image at least four times (display the original and at least three modifications). **Your program must be saved in a file called “hw1.py” in your csis110/hw/hw1 folder.** You can use any image you want, something you find on the web or a picture of your dog, or whatever – just make sure it’s not the same picture anyone else is using, to your knowledge. The first image you display must be the original, unaltered image. The others must be modified forms of the original image. You can use any of the manipulations we used in lab or that are described in Chapter 4 of the Python book, you can shift or alter colors, make it darker/lighter, or virtually anything else you want. Be creative and have fun with it!! Your program must also pause after each version of the image is displayed, to give the user time to appreciate your work. Finally your program must print the width and height of the image (in pixels) and the number of pixels in the image.

We have given you a working program that does part of this so that you have a starting point: it is printed below. We are not giving you this program online, only on paper, so you will need to type it in.

**NOTE**: A perfectly working program as described above will earn 30 points. To earn the remaining 5 points, you must use your own creativity to go above and beyond those minimal requirements. Examples include additional manipulations of the original picture, original or unique or difficult manipulations, etc. Go nuts!

**Tips**:

1. Type in the working program that we give you below. Get that to work in your account, then change it to use a different picture, add more manipulations, etc.
2. Build on what you’ve learned! There’s actually not much that’s new here, mostly putting together things you’ve already done. If you don’t have your book beside you and your code from the labs in front of you when you’re doing this, then you are going to waste a lot of time.

**TURNING IT IN:**

1. Turn in a printout of your code (the hw1.py file) with the rest of your homework.
2. Put a copy of your hw1.py file and your original image in your csis110/hw/hw1 folder. THIS IS REQUIRED IN ORDER FOR ME TO GRADE IT AND FOR YOU TO EARN FULL CREDIT.

# CSIS110, HW1, Spring 2018, sample code and template

#

# We're giving you this to help you get started. Type this in and

# get it to run, THEN change the picture to one of your choosing

# and complete the rest of the program as required. Make sure your

# program fulfills all the assignment requirements, and have fun!!

# Program creates a sequence of 2 copies of the "bigben.jpg" picture,

# the original plus one with less red.

#

# To run this program: put the picture you manipulate into your

# hw/hw1 folder, load it into JES, run "setMediaPath()" from command

# area to set the media path to your hw/hw1 folder, then

# type "createSequence()".

import time

# Reduce red in picture by 50% (code from book, Program 34)

def decreaseRed(picture):

 for pix in getPixels(picture):

 value = getRed(pix)

 setRed (pix, value \* 0.5)

# Creates a sequence of 2 versions of an image, the original

# then one with less red. Also reports the width of the image.

def createSequence():

 # Find original picture (you will use a different picture)

 myPic = makePicture (getMediaPath( "bigben.jpg"))

 # Display the picture and pause 3 seconds

 repaint (myPic)

 time.sleep(3)

 # Perform the first manipulation of the picture and display it

 decreaseRed (myPic)

 repaint (myPic)

 # other manipulations and requirements go here, after you get the

 # initial program working ....

 # wrap up: print image statistics

 width = getWidth (myPic)

 print "this image is " + str(width) + " pixels wide"

# CSIS110, HW1, Spring 2018, example solution.

# This would be worth 30 / 35 points, because

# it uses very simple functions for image

# manipulation, without trying to get too creative.

#

# To run this program: put the picture you manipulate into your

# hw/hw1 folder, load it into JES, run "setMediaPath()" from command

# area to set the media path to your hw/hw1 folder, then

# type "createSequence()".

import time

# Reduce red in picture by 50% (code from book, Program 34)

def decreaseRed(picture):

 for pix in getPixels(picture):

 value = getRed(pix)

 setRed (pix, value \* 0.5)

# increase blue value of every pixel by 50%

def increaseBlue(picture):

 for pix in getPixels(picture):

 value = getBlue(pix)

 setBlue (pix, value \* 1.5)

# turn an image into its negative (book, Program 41)

def negative (picture):

 for px in getPixels (picture):

 red = getRed (px)

 green = getGreen (px)

 blue = getBlue (px)

 negColor = makeColor (255-red, 255-green, 255-blue)

 setColor (px, negColor)

# Creates a sequence of 4 versions of an image, the original

# then three more as described above. Also reports the dimensions

# of the image and its total number of pixels.

def createSequence():

 # Find original picture (you will use a different picture)

 myPic = makePicture (getMediaPath( "bigben.jpg"))

 # Display the picture and pause 3 seconds

 repaint (myPic)

 time.sleep(3)

 # Perform the first manipulation of the picture and display it

 decreaseRed (myPic)

 repaint (myPic)

 time.sleep(3)

 # Perform the second manipulation of the picture and display it

 increaseBlue (myPic)

 repaint (myPic)

 time.sleep(3)

 # Perform the third manipulation of the picture and display it

 negative (myPic)

 repaint(myPic)

 # wrap up: print image statistics

 width = getWidth (myPic)

 height = getHeight(myPic)

 print "this image is " + str(width) + " pixels wide"

 print "this image is " + str(height) + " pixels high"

 print "this image contains " + str(width\*height) + " pixels"