Andre Martin
CSE – 31 – Lab 2

TPS activities/assignments

TPS (Think-Pair-Share) Activity 1:

1. Independently search from the internet 3 online tutorials on how to setup and use GDB in your system.
   - I Install Homebrew on the Homebrew Website.
   - How to Use the brew command line: brew install GDB
   - How to set up GDB on a Mac.com
   - I shared with my partner, I've bookmarked it and we discussed it with our TA at random.

TPS activity 2:

1. How do you compile your ***punishment.c*** so that you can debug it using GDB? Try it with your code and make the name of the executable ***punish***.
   - On Mac GDB doesn't work so I use the LLDB bugger on X Code.
2. Once ***punishment.c*** is compiled, how do you load it in GDB? Try it with your program.
   - To load it onto x code I create new directory, I insert punishment.c and run it with the play button on x code.
3. Once ***punish*** is loaded, how do you run it in GDB? Try to run you ***punish***.
   - To run it with LLDB on x code you go to the Debug Session tab.
4. What are breakpoints? How do you set a breakpoint at a certain line of your program? Try to set a breakpoint in ***punishment.c*** where the ***for loop*** begins.
   - Breaking points are used to inspect the current state of program exeution, we need to break the execution of debugging program in gdb
   - To set a break point at a certain line in x code you press the number of the line where you want to put the breakpoint.
5. Now run the program again. It should stop at the breakpoint you set in Q4. From here, how do you run the program line by line? Try to run the next 3 lines with your program.
   - To run the program line by line after the break point with x code you use the step over command.
6. While you are still running ***punish*** line by line, how can you see the value of a variable? Pick 3 variables in your program and display them in the terminal one by one.
   - Yes, you can still see the value of the variable.
7. Now you are tired of running line by line. How do you let the program finish its run? Try to finish running your ***punish***.
   - On mac you press the continue program execution on X code.
8. How do you exit from GDB?
   - To exit LLDB on Mac you just finishing running the program or delete the break point.

TPS activity 3:

1. How many variables were declared in line 1? How many of them are pointers (and what are they)?
   - 5 variables are declared in the program. x and y are integers. *px and *py are pointers to integers and arr is an array of 10 integers.
2. What will be the values of ***x***, ***y***, and ***arr[0]*** if you run the program? Validate your answer by running the program. Why do you think it happens that way? You will need to insert print statements to display those values.
   - The program doesn't run, it says Program ended with exit code: 0, this happens because you haven't inserted the print statements. x,y, arr[0] can have some random numbers.
3. How do you prevent ***x***, ***y***, and the content of ***arr*** from having unexpected values? Try to fix them in the program.
   - To prevent x,y and arr from having unexpected values you need to insert print statements. Also, x,y and arr[0] can have some random values because they are uninitialized. To avoid this, initialize them.
4. The moment you have declared a variable, the program will allocate a memory location for it. Each memory location has an *address*. Now insert print statements to display the ***addresses*** of ***x***, ***y***.
   - Yes, I inserted the print statements to display the addresses on x and y.
5. Now insert code so that ***px*** points to ***x*** and ***py*** points to ***y***. Print out the values and addresses of those pointers using only the pointer variables (yes, pointers have addresses too!) You should see that value of ***px*** equals to address of ***x***, and the same is true with ***py*** and ***y***.
   - Yes, I inserted the code and px equals to address x and py equals address y.
6. As we've learned in lectures, an array name can be used as a pointer to access the content of the array. Write a loop to print out the content of ***arr*** by using ***arr*** as a pointer (**do not use []**).
   - I wrote the loop to print out the content of arr by using arr as a pointer.
7. Are array names really the same as pointers? Let's find out! An array name points to the first element of an array, so ***arr*** should point to the address of ***arr[0]***. Insert code to verify this.
   - I inserted the code to verify this and it did.
8. Now print out the address of ***arr***. Does the result make sense? Why?
   - Yes, the results make sense because it will result in the address of array. A large number.

(Assignment 1, individual) Segmentation Faults

Answer the following questions:

1. What line caused the segmentation fault?
   - Line 19 this code scanf("%d",input); in while loop in function is causing segmentation fault since it assumes input as address
2. How do you fix the line, so it works properly?
   - It fix the line so it can work properly you input scanf("%d",&input) ;
3. What is the big here?
   - Average is not calculated properly since whatever is changed in function is local to the function it can't be accessed in main.
4. How do you fix it?
   - Pass it by reference as argument in function

(Assignment 2, individual) Fix appendTest.c

1. Run the program with the following input: ***"HELLO!"*** for **str1** and ***"hello!"*** for **str2**. Is the output expected?
   - Its normal and print as HELLO!hello!
2. Do not stop the program, enter ***"HI!"*** for **str1** and ***"hi!"*** for **str2**. Is the output expected? What is the bug here? Try to fix the program so it will print the output correctly.
   - there is end of line in char * .. the previous text content is displayed.
   - to fix it…..s1[s1len + s2len] = '';
3. Do not stop the program, enter ***"Hello! How are you?"*** for **str1** and ***"I am fine, thank you!"*** for **str2**. Is the output expected? Why do you think this happens? **You don't need to fix this**.
   - char str1[10];
     char str2[10];
   - The size str1 and str2 are 10 ... and you are entering more then 10 the program would not accept it.