lCMPS 250 Students,

In an effort to increase understanding and to provide a further opportunity for each of you to demonstrate your understanding of the more challenging questions from the recent Mid-Term (and their answers), I’m inviting you to formulate answers to each of the following questions and to submit them to the MidTerm folder on the CWS (by the end of the day on Mon Mar 28th). Specifically, you at to submit each of the following:

1. Your completed version of the C program for VI, VI.c. Be sure to place the appropriate identifying comments at the top of the source program.
2. Your completed version of the C program for VII, VII.c, again with the appropriate identifying comments at the top.
3. Your completed version of the PEP/8 program for VIII, VIII.pep, also with the appropriate identifying comments at the top.
4. Your revised version of this file, MidTermPostS2016.docx, with your name in the header and with your answers clearly inserted into the document at each of the appropriate places, along with accompanying screen shots for each illustrating that that code works.

Furthermore, don’t be surprised if we spend some of our upcoming class time during that week with you individually answering **similar** questions and submitting those answers. It is my hope that the above activity will better prepare you to provide overall better and more complete answers to such similar questions.

Good luck,

P.M.J.

Answers are Highlighted

1. Consider the following two equivalent assembly language programs (*for 32-bit Intel AT&T Syntax and PEP/80*); they both perform the same task and will produce the same printed results. As your answer to this question you are to provide a complete C source program that is likewise equivalent. (15%/75%)

.text

.globl main

.align 16, 0x90

.type main,@function

main:

pushl %ebp

movl %esp, %ebp

subl $24, %esp

leal format, %eax

movl $0, -4(%ebp)

movl $1000, -8(%ebp)

movl $1, -12(%ebp)

movl $1, -16(%ebp)

movl -12(%ebp), %ecx

movl %eax, (%esp)

movl %ecx, 4(%esp)

calll printf

leal format, %eax

movl -16(%ebp), %ecx

movl %eax, (%esp)

movl %ecx, 4(%esp)

calll printf

loop:

movl -12(%ebp), %eax

addl -16(%ebp), %eax

movl %eax, -20(%ebp)

movl -20(%ebp), %eax

cmpl -8(%ebp), %eax

jg done

leal format, %eax

movl -20(%ebp), %ecx

movl %eax, (%esp)

movl %ecx, 4(%esp)

calll printf

movl -16(%ebp), %ecx

movl %ecx, -12(%ebp)

movl -20(%ebp), %ecx

movl %ecx, -16(%ebp)

jmp loop

done:

movl -4(%ebp), %eax

addl $24, %esp

popl %ebp

ret

format: .asciz "%d\n"

; CMPS 250 - Spring 2016, Mid-Term, VI.pep

BR main

w: .WORD 1000

x: .WORD 1

y: .WORD 1

z: .BLOCK 2

main: DECO x,d

CHARO '\n',i

DECO y,d

CHARO '\n',i

loop: LDA x,d

ADDA y,d

CPA w,d

BRGT done

STA z,d

DECO z,d

CHARO '\n',i

LDA y,d

STA x,d

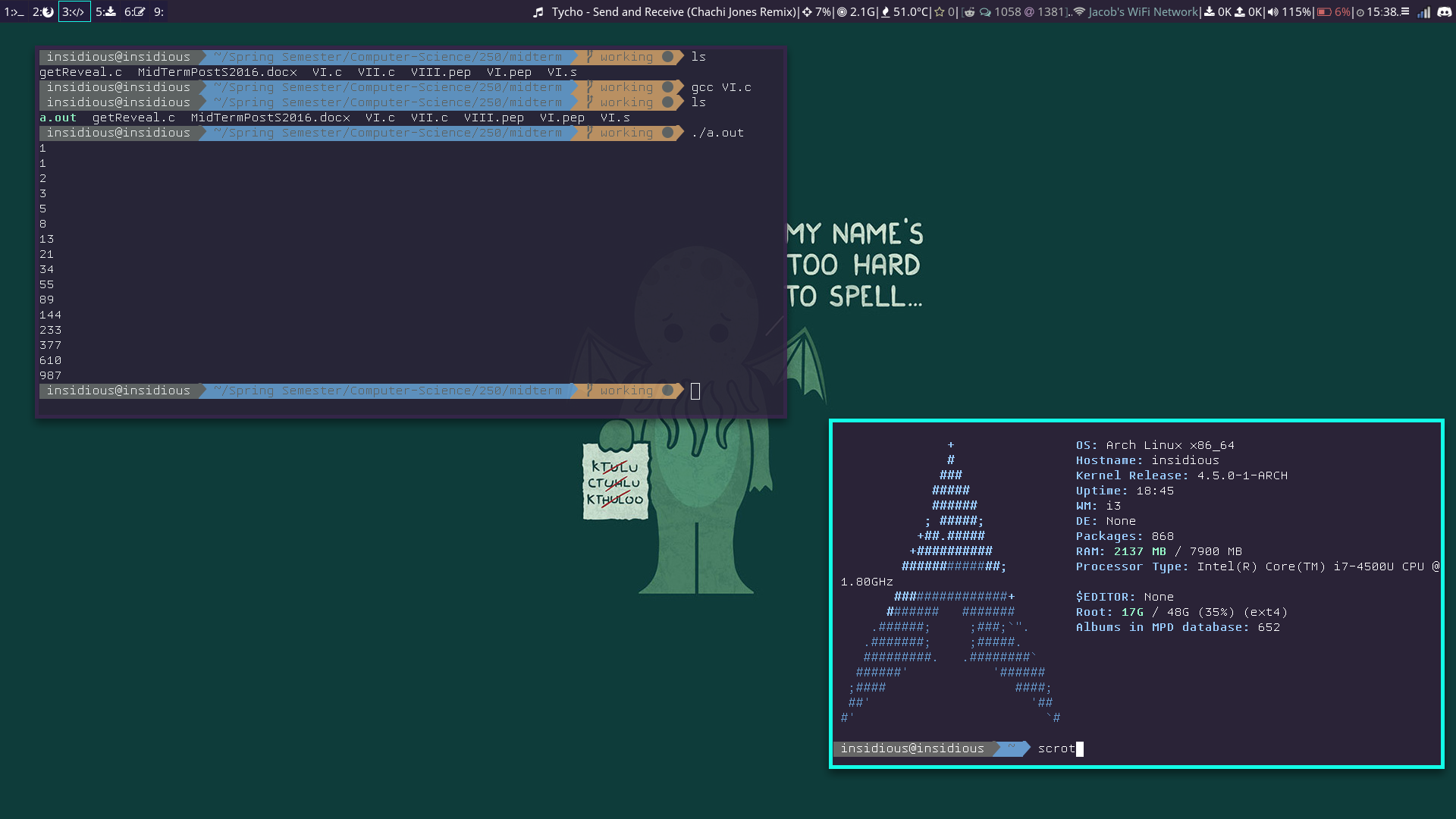
LDA z,d

STA y,d

BR loop

done: STOP

.END



#include <stdlib.h>

#include <stdio.h>

int main(){

int w = 1000;

int x = 1;

int y = 1;

int z;

printf("%d%s", x, "\n");

printf("%d%s", y, "\n");

while(x+y < w){

z = x+y;

printf("%d%s", z, "\n");

x = y;

y = z;

}

}

1. Consider the following C subprogram that replaces every instance of the given character c by the given character replacement in the string passed as the first argument.

#define NC ‘\0’

void replaceAll(char \*array, char c, char replacement) {

int index = 0;

while(array[index] != NC) {

if(array[index] == c) {

array[index] = replacement;

}

index = index + 1;

}

}

As your answer to this question you are to provide an implementation for the following C subprogram that “removes” every instance of the given character c from the string passed as the first argument. Thus, "Paul" becomes "PXul" with ‘a’ **replaced** by ‘X’ and "Pul" with ‘a’ **removed**. Similarly, “abracadabra” becomes "XbrXcXdXbrX" and "brcdbr" respectively. (15%/90%)

void removeAll(char \*array, char c) {

int count = 0;

for(int i = 0; i < strlen(array); i++){

if(array[i] == c){

count++;

}

}

char \*result = (char\*)malloc(strlen(array)-count);

int j = 0;

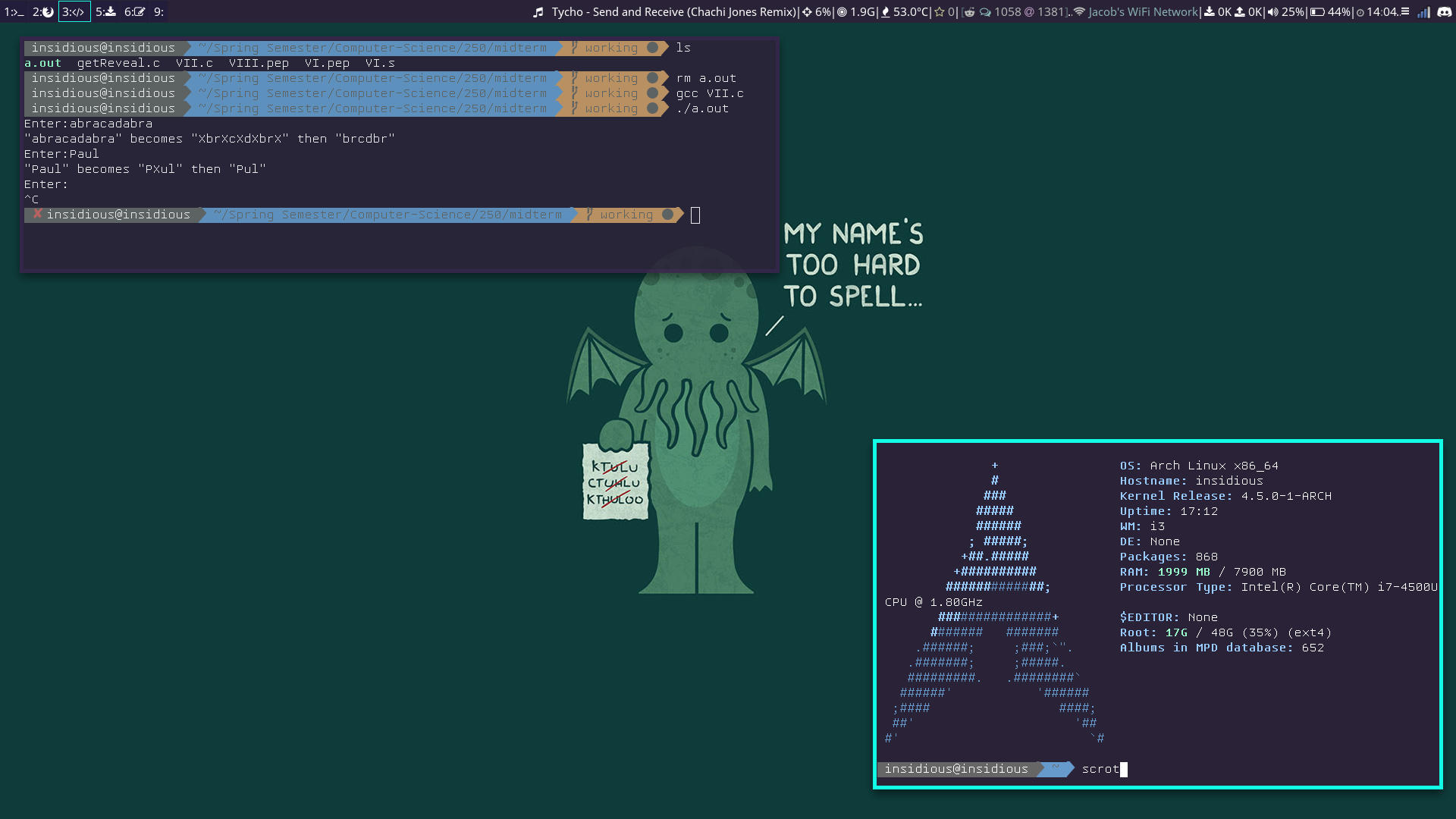
for(int i = 0; i < strlen(array); i++){

if(array[i] == c){

}else{

result[j] = array[i];

j++;

}

}

strcpy(array, result);

free(result);

}

What follows is a simple Pep/8 program that is nearly complete. This program is designed to read two integer values as input and then determine and print the sum of the integers in the inclusive interval defined by these two values. For this question you are to provide the missing statements to complete the program. (10%/100%)

;P.M.J. (Spring 2016)

;A program to determine the sum of integers in an inclusive interval

;

**1**

**5**

15 is the sum.

BR main

message: .ASCII " is the sum.\n"

.BYTE 0x00

value1: .BLOCK 2

**-2**

**6**

18 is the sum.

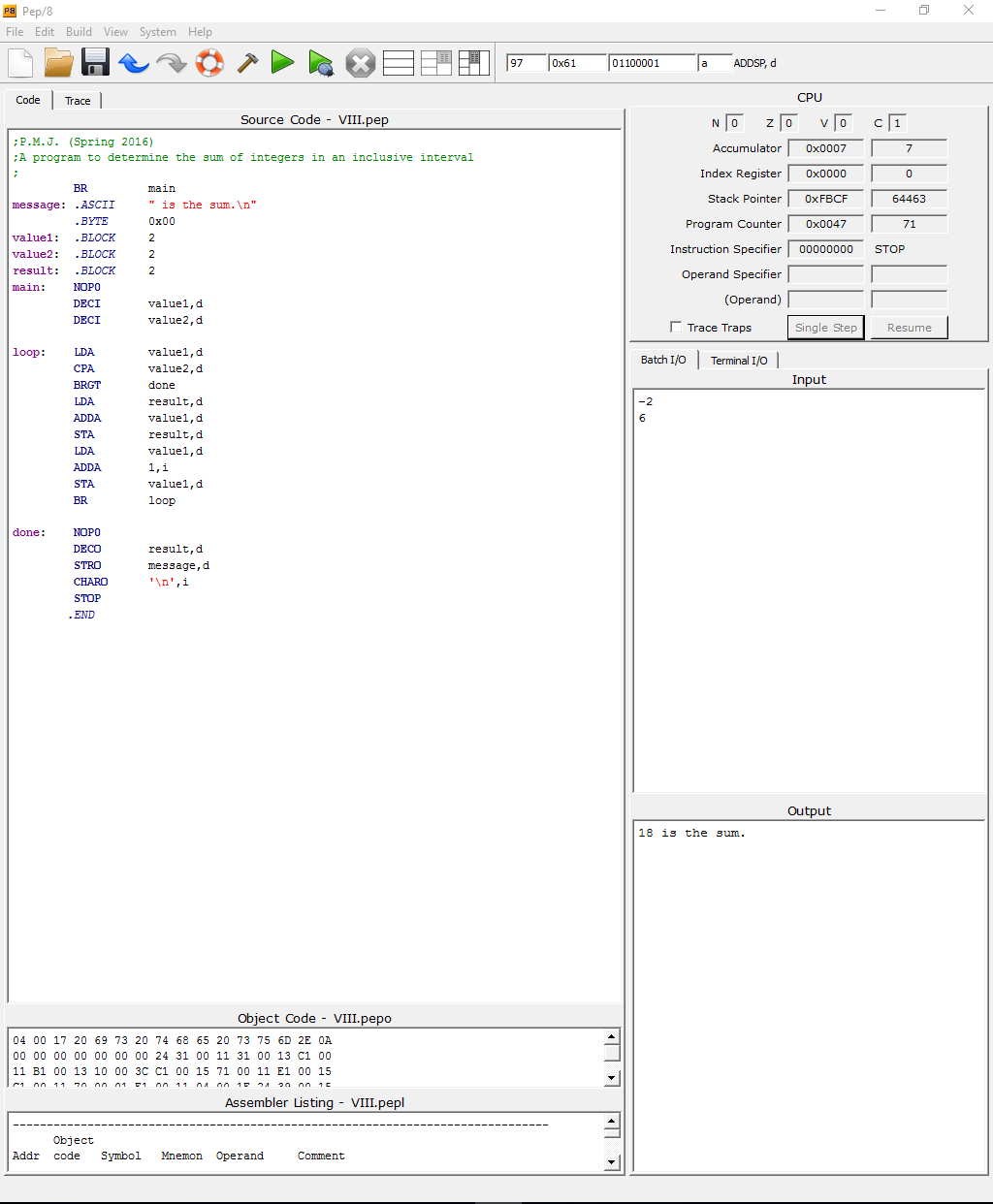
value2: .BLOCK 2

result: .BLOCK 2

main: NOP0

DECI value1,d

DECI value2,d



loop: LDA value1,d

CPA value2,d

BRGT done

LDA result,d

ADDA value1,d

STA result,d

LDA value1,d

ADDA 1,i

STA value1,d

BR loop

done: NOP0

DECO result,d

STRO message,d

CHARO '\n',i

STOP

.END