Exercise 1: Sort the output from smallest to the largest.



#include <stdio.h>



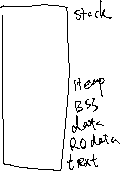
#include <stdlib.h>

int x = 20;

int y;



int



foo(){ }

int



main(){

int z = 20;

int \* ptr = (int\*)malloc(2\*sizeof(int));



printf("%p\n", &x); //A



printf("%p\n", &y); //B

printf("%p\n", foo); //C



printf("%p\n", &z); //D

printf("%p\n", ptr); //E



free(ptr);

}



Exercise 2: Are the following codes ok?



char \*bytes = malloc(1024 \* sizeof(\*bytes));

char \*ptr = "cse30";

/\* some code \*/

free(bytes + 5); **// A. Yes B. No**



free(ptr); **// A. Yes B. No**



Exercise 3: Which one of the following may cause a dangling pointer? How about memory leak?

void foo(int bytes) {



char \*ch =(char \*) malloc(bytes);



. . . . //unrelated to ch

}

======================================

int \*

foo(int bytes){

int i=14;

return (&i);

}



int

main () {

int \*p = foo(10);

}

======================================

char\* foo(int bytes) {



char \*ch =(char \*) malloc(bytes);



return (ch);



}



======================================

char \*str = strdup("POINTERS…”);



\*str = 'h';



str = NULL;



======================================

int \*ptr = (int\*)calloc(5, sizeof(int));

int \*end = ptr+5;

while(ptr < end){

printf("%d, ", \*ptr++);



}

free(ptr);

