CS 240: Programming in C Midterm 2 Spring 2025

Practice Midterm 2

Version I

Name:	
Username:	

Read all instructions before beginning the exam.

- This exam is intended to be equally or more difficult than the past official midterm provided.
- You are encouraged to post on Ed Discussion, without hesitation, any sort of question you may have about this practice exam.
- This is a closed book examination. No material other than those provided for you are allowed.
- You need only a pencil and eraser for this examination. If you use ink, use either black or blue ink. If you use pencil, your writing must be dark and clearly visible.
- This examination contains an amount of material that a well-prepared student should be able to complete in less than one hour.
- This examination is worth a total of 100 points. Not all questions are worth the same amount. Plan your time accordingly.
- Write legibly. You should try to adhere to the course code standard when writing your solution(s). Egregious violations may result in point deductions.
- Read each question carefully and only do what is specifically asked for in that problem.
- Assume appropriate includes have been added to the code segments shown in the problems.
- Circle your answer in true or false questions.
- Some problems require several steps. Show all your work. Partial credit can only be rewarded to work shown.
- Write your username on EVERY page where indicated. Any page without a username will receive a zero for the material on that page.

Signature:

Do not open the examination booklet until instructed.

- 1. (30 points) Write brief answers to the following questions.
 - (a) (3 points) What is the output of the following program? State the value displayed by printf(), undefined behavior, or error produced. Assume no program termination occurs due to assertions.

```
typedef struct {
    char *text;
} line_t;

int main() {
    line_t *record = malloc(sizeof(line_t));
    assert(record);

    if (!record->text) {
        record->text = malloc(15);
        assert(record->text);
        strcpy(record->text, "Dennis Ritchie");
    }

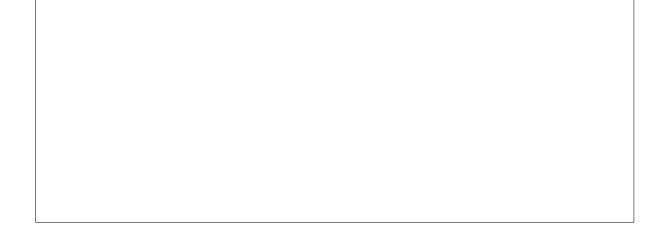
    printf("%s\n", record->text);
}
```

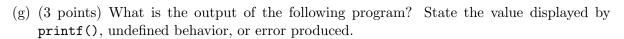
(b) (3 points) The code segment in problem 1.a. has memory leaks. State the line(s) of code to be implemented in order, at the end of main(), to prevent them from occurring.

Username:

- (c) (1 points) True or False: Considering a singly linked list, the expression head + 1 is always functionally equivalent to head->next.
- (d) (5 points) State whether the variables in the following code segment <u>point</u> to a section in the heap, stack, neither, or if it is undefined behavior, at the end of the function.

- (e) (1 points) True or False: The function in the code segment in problem 1.d. will produce memory leaks when called.
- (f) (2 points) Briefly explain what the bt GDB command does.





```
void modify(int **ptr) {
    int y = 20;
    *ptr = &y;
}

int main() {
    static int x;
    int *p = &x;
    int **pp = &p;

    modify(pp);
    printf("%d\n", x);
}
```

(h) (3 points) Briefly explain the problem that may arise from calling free() on a memory block that has already been deallocated.

1		
1		
1		
1		
1		
1		
1		
1		
1		
1		
1		
1		

Username: Midterm 2 (Practice) Spring 2025

 $\mathrm{CS}\ 240$

(i)	(3 point) Write the function prototype of func that takes in an integer and a constant
	string as arguments, and returns a function pointer that takes in and returns a struct
	node pointer.

- (j) (1 point) True or False: A function with the signature void func(const int *p); guarantees that the value pointed to by p will never change throughout the program.
- (k) (5 points) In the code segment, there are 6 different arrow operators (->). Rewrite the function, replacing all arrow operators for pointer implementations (*).

```
void process(node_t *head) {
    while ((head) && (head->next)) {
        head->data += head->next->data;
        head = head->next->next;
    }
}
```

Username:	Midterm 2 (Practice)	CC 240
	Spring 2025	CS 240
(a) (4 points) Declare a sand a float named by	questions deal with dynamic allocation as structure named flight containing a clooarding. The structure should be a value a type flight_t that refers to it.	haracter pointer named id,

(b) (4 points) The following code segment attempts to create a newly dynamically allocated linked list node of the structure created previously, populating it with argument values, where id must be not NULL, and boarding must be non-negative. There exists one problematic behavior in the code segment. Briefly explain what the problem is and propose a rewrite of one existing non-blank line to fix it.

```
flight_t *create_node(char *id, float boarding) {
2
       assert(id);
3
       assert(boarding >= 0.0);
4
5
       flight_t *new = malloc(sizeof(flight_t));
6
       assert(new);
7
8
       new->boarding = boarding;
9
       new->id = malloc(strlen(id) + 1);
10
       assert(new->id);
11
       strcpy(new->id, id);
12
13
       return new;
14 }
```

Username:						Midterm 2 (Practice)	
						Spring 2025	CS 2

240(c) (12 points) Write a function, swap_flights() that takes two arguments, the address of a pointer to a flight_t node named fA, and the address of a pointer to a flight_t node named fB. The two arguments reference pointers pointing to two flight nodes already in the linked list. Rearrange the linked list so that these two nodes exchange positions in the list without swapping their data fields. The function's return type is void. Use assertions where necessary. You may assume that the arguments contain the address of the previous node's next pointer, or the head pointer if the first node is being replaced.

Use	rn	am	e:			Midterm 2 (Practice)
						1 (1 ractice)
						Spring 2025
						1 0

ontinuation of 2.c.

 ${\rm CS}\ 240$

						Spring 2025	CS 240
(d)	tion of the control o	of b he twist) ype hot irst hece	sing oard oard oard oard oard oard oard oard	ly l	inke g va umer fun integ d. cond the l ent : An e	te a function, inspect_flights(), to perform a single d list, swapping every two nodes in order to maintain lues, where the head of the list has the smallest value. Its, the address of the pointer to a flight_t node (the ction pointer to the swap_flights() function in problem argument to swap flight nodes. Make sure to call the ist remains a valid singly linked list, and that the postill references the head of the list upon completion. Using the list should not fail an assertion. Remember you sorted list, as you only need to perform one linear travelat swap_flights() was implemented appropriately.	an ascending order The function takes head of the linked em 2.c. The return dified, or 0 if it was e function correctly, hinter passed in the Use assertions where are not expected to

Us€	ern	am	e:			Midterm 2 (Practice)
						17114001111 2 (114001100)
						Spring 2025
						T O

ontinuation of 2.d.	

 ${\rm CS}\ 240$

Us	ernai	me:						Μ	Iidtern Spr	n 2 (1 ring 2		ce)						CS 240
3.	(40	poir	nts) T	Γhe	foll	owing	g quest	ions c	leal wi	ith d	ynam		catio	n and	bina	ry tre	es.	
	(a)	(3 points) Declare an enumerated type named type that contains two named constants INT and PAIR, in that order. Simultaneously create a type type_t that refers to it.																
	(b)	lan	guag es: i	e. nte	Rep	resen	t the c	object A pa	types ir can	of a be a	mini	mal p	rogra	mmir	ıg lar	nguag	e that	ramming has two in integer
		For the $\frac{\sinh a}{\sinh a}$	this sam red r stru	s, a ne s ner	struc nory hold	uctur ture spac s an	e can type (e, as to integer	be de the pa o repre r or a	eclared air of esent e	tha poin	ters v an in	vrappe teger (ed to or a p	gethe pair ty	r in a	anothespect	er str ively.	to two of uct) in a Whether umerated
		Dec cha and sho the	clare racte l a p uld l inte	a ser n pair be	structiame of wrap . Sin	cture ed ma: strue oped nulta	rk, and et obj around neously	d objed, for ject plant anot y crea	the stepointer ther in the a ty	orage rs na iterna ype c	e of the med al stru bject	e obje pair_i ict an	ect's firs d sha	value, t and are th	an i pair e san	ntegei c_sec	nam	unsigned ed value The pair space as
		Hir	nt: U	$\frac{se}{-}$	a un 	nion i	nside o	of the ——	о <i>в</i> јесt 	stru	<i>ct</i> .							

		Spring 2025	CS 240
Continu	ation of 3.b.		

Username:						Midterm 2 (Practice)	
						Spring 2025	CS 240

	1 0
(c)	(8 points) Write a function, new_object(), which accepts two parameters, a type_t enum, and an integer. The first argument determines the type of the object being created, and the second argument corresponds to the integer value that the new object holds. The second argument must only be used if the first argument indicates that the object is an integer type (INT), otherwise it must be ignored. Return a pointer to a newly allocated object_t object. Assert for the first argument to be valid, and assert in other places where it may be necessary (do not assert the second argument). Be sure to initialize all fields.

Username:						Midterm 2 (Practice)	
						Spring 2025	CS 240

(d) (10 points) Pair type objects provide two references to two other objects, that may or may not be of the same type. This layout can mirror a binary tree, where the left and right child are the pair_first and the pair_second pointers respectively, turning the associated objects into nodes. Write a recursive function named set_marks() that takes one argument, a pointer to an object_t node, the root of the objects' reference tree. Traverse the tree in postfix order, and set the mark character to 1 in each object node. Return an integer of the number of nodes that were modified. Consider that some nodes may already have mark set to 1 in which case they should not be counted as modified nodes.

Username:							Midterm 2 (Practice)	
							Spring 2025	CS 240

	5pmg 2020 05 240
(e)	(12 points) Write a recursive function named sweep_marks() that takes one argument, a pointer to the address of an object_t node, the root of the objects' reference tree. For every node that has mark set to 0, set all of its children nodes' mark to 0. This sweeping operation should end up with all nodes emerging from a parent node, that has its mark set to 0, until the associated leaf nodes, with their mark also being set to 0. Additionally deallocate, maintaining a valid binary tree structure, all node objects of type INT that have their mark set to 0 by the end of the sweeping operation. Return the integer sum of the deallocated nodes' values.

Username:	Midterm 2 (Practice)	
	Spring 2025	CS 240

Continuation of 3.e.	