PIC 40A Midterm

CHARLES ZHANG

TOTAL POINTS

29 / 32

QUESTION 1

1 When type==object, value amounts to object reference 3 / 6

(a)

✓ **+ 3 pts** *Correct.*

(b)

✓ **+ 0 pts** *Incorrect.*

QUESTION 2

2 IIFEs, function arguments are passed by value, lexical environments introduced by function calls 8 / 8

(a)

✓ **+ 2 pts** *Correct.*

(b)

✓ **+ 2 pts** *First line is correct.*

✓ **+ 2 pts** *Second line is correct.*

✓ **+ 2 pts** *Third line is correct.*

QUESTION 3

3 Prototypal inheritance 6 / 6

✓ **+ 2 pts** *(a) is correct.*

✓ **+ 2 pts** *(b) is correct.*

✓ **+ 2 pts** *(c) is correct.*

QUESTION 4

4 Classes, function arguments are passed by value 4 / 4

✓ **+ 4 pts** *Correct.*

QUESTION 5

5 Event handlers, (), this 8 / 8

First (a)-(d).

✓ **- 0 pts** *Correct.*

Second (a)-(d).

✓ **- 0 pts** *Correct.*

Excellent! The only student in the class to get this (pun intended) correct.

gradescope

PIC 40A
Introduction to Programming for Internet

Midterm A

Instructions:

- You have **50 minutes** to complete the exam.
- There are **5** problems (1ab, 2ab, 3a-c, 4, 5(a-d)×2) worth a total of 32 points.
- You may **not** use any books or notes.
- Write your solutions in the space **below** or **next to** the questions and **box what you want me to read** clearly.
- If you need space for thinking, ask for **scratch paper**.
- Do not forget to write your name and UID in the space below.

Name: Charles Zhang

Student ID number: 305413659

Question	Points	Score
1	6	
2	8	
3	6	
4	4	
5	8	
Total:	32	

Problem 1. *6pts.*

In each part...

- **Say if an error is encountered during code execution.**
- **If there is no error, write down the output from executing the code.**

For concreteness, you can assume that you're running the code in JSFiddle. In particular, you don't have to worry about how an advanced JS console (like Chrome's) would display the output.

a -> [0]
b -> [1,2]
c ->

(a)
```
const a = [];
const b = [];
const c = b;

a.push(0);
b.push(1);
c.push(2);

console.log(a);
console.log(b);
console.log(c);
```

[0]
[1,2]
[1,2]

arr -> ~~[0]~~ ~~[0,[0]]~~ ~~[0,[0,[2]]]~~ [0,[0,[2],3]] [0,[0,[2],3],4]

(b)
```
let arr = [0];

arr.push(arr);
arr[1].push([2]); // we're pushing [2], not 2
arr[1].push(3);
arr.push(4);

console.log(arr[1][1][0]);
console.log(arr[1][2]);
console.log(arr[2]);
```

Error, pushing arr into itself creates a circular reference

Problem 2. *8pts.*

In each part...

- **Say if an error is encountered during code execution.**
- **If there is no error, write down the output from executing the code.**

For concreteness, you can assume that you're running the code in JSFiddle. In particular, you don't have to worry about how an advanced JS console (like Chrome's) would display the output.

(a)

```
(function(f, N) {
  let sum = 0;

  for (let i = 1; i <= N; ++i) {
    sum += f(i);
    console.log(sum);
  }

})(function(i) { return 2 * i; }, 3);
```

(b)

```
function f(a1) {
  let a2 = [];

  (function() {
    a1.push(0);
    a2.push(0);
    a3.push(0);

    a1 = [1];
    a2 = [2];
    a3 = [3];
  })();
}

let a1 = [];
let a2 = [];
let a3 = [];

f(a1);

console.log(a1);
console.log(a2);
console.log(a3);
```

Problem 3. *6pts.*

Consider the following code that executes without encountering an error.

```
let o1 = {
  b: 1,
  c: 1,

  f: function() { console.log(this.a); },
  g: function() { console.log(this.b); },
  h: function() { console.log(this.c); }
};

let o2 = {
  a: 2,
  b: 2
};

let o3 = {
  h: function() { console.log(this.c + 3); }
};

Object.setPrototypeOf(o2, o1); // set o2's prototype to be o1
Object.setPrototypeOf(o3, o2); // set o3's prototype to be o2
```

o3 → o2 → o1

o3 → {a:2, b:2, c:1, o1.f, o1.g, o3.4}

In each part...

- **Say if executing the specified additional code leads to an error.**
- **If there is no error, write down the output from executing the code.**

(a) `o3.f();`

2

(b) `o3.g();`

2

(c) `o3.h();`

4

Problem 4. *4pts.*

Consider the following code that executes without encountering an error.

```
function Rectangle(sl1, sl2) {
  this.sideLength1 = sl1;
  this.sideLength2 = sl2;
}

Rectangle.prototype.area = function() {
  return this.sideLength1 * this.sideLength2;
};


function changeToSquareOfSameArea(r) {
  let commonSideLength = Math.sqrt(r.area());
  r.sideLength1 = commonSideLength;
  r.sideLength2 = commonSideLength;
}

function stretchByTwo(r) {
  r = new Rectangle(2 * r.sideLength1, 2 * r.sideLength2);
}

let r = new Rectangle(2, 8);

changeToSquareOfSameArea(r);
stretchByTwo(r);

console.log(r.sideLength1 + ' x ' + r.sideLength2);
```

Write down the output from executing the code.

4 x 4

Problem 5. *8pts.*

A student in PIC 40A writes the following HTML.

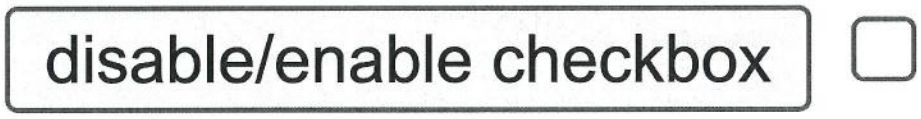
```
<!DOCTYPE html>
<html lang="en">

  <head>
    <meta charset="UTF-8">
    <title>PIC 40A Testing</title>
    <script src="testing.js" defer></script>
  </head>

  <input type="button" value="disable/enable checkbox" id="btn">
  <input type="checkbox" id="xbox">

</html>
```

The HTML validates and when no code is written in the file `testing.js` it produces the following (minimal) webpage.



The student then writes the following code in `testing.js`.

```
const button = document.getElementById('btn');
const checkbox = document.getElementById('xbox');

checkbox.toggle = function() {
  this.disabled = !this.disabled;
};

button.addEventListener('click', checkbox.toggle());
```

The student wants their button to do what it says: toggle the checkbox between enabled and disabled.

(a) Does their webpage perform as they want it to? (yes/no)

(b) What is the status (enabled/disabled) of the button and checkbox after the browser has parsed the HTML and executed the JS?

Button: enabled
Checkbox: disabled

(c) What does clicking on the button accomplish the **first** time it is clicked?

Nothing, the listener is executing the value of checkbox.toggle(), which is undefined

(d) What does clicking on the button accomplish the **second** time it is clicked?

Nothing, same as (c)

Dissatisfied, the student removes the parentheses from the final line of JS so that it says

```
button.addEventListener('click', checkbox.toggle);
```

and reloads the page.

(a) Does their webpage perform as they want it to now? (yes/no)

(b) What is the status (enabled/disabled) of the button and checkbox after the browser has parsed the HTML and executed the JS?

Button: enabled
Checkbox: enabled

(c) What does clicking on the button accomplish the **first** time it is clicked?

Disables the button, since "this" is being used in an event listener, it binds to the button, not the checkbox

(d) What does clicking on the button accomplish the **second** time it is clicked?

Nothing, it's disabled

THIS PAGE IS INTENTIONALLY LEFT BLANK