EECS 368
Sp 2018
Midterm

~~17.25~~ +1 18.25 Name: student Number: 2856[illegible]

Write your name on every page. Accidents happen.

True/False. If the answer is false, correct the statement so that it is true.

(20 pts)

T (F)

1. The following code will output "no.":

```
let x = true;
function someFunction() {
    if(x) {
        console.log("yes");
    }else {
        console.log("no");
    }
    let x = false;
}
someFunction();
```

False
Reference error, because there are temporal dead zone before define x; ~~we cannot~~

+2

2. a is at a higher level of abstraction than b.

b is higher level of abstraction than b. ~~False~~ action is a high-order function in b;

```
a.functionrepeatLog(n){
    for(let i=0;i<n;i++){
        console.log(i);
    }
}
```

```
b.functionrepeat(n,action){
    for(let i=0;i<n;i++){
        action(i);
    }
}
```

+2

3. will

```
if(x={}){
    console.log(1);
}
```

output '1'.

TRUE

+2

4. In JavaScript, using == to compare different **objects** with the exact same properties will return `false` but using === to compare the same two objects will return `true`.

False. JavaScript has no deep comparsion! === to compare same object is ~~nul~~ return false!

+2

5.

```
let x = {info: "Important!"};
let y = x;
x.info = "Not Important!";
```

`x.info === y.info` will evaluate to `true`

TRUE

+2

6. JavaScript is based on the Java Programming Language

False! JavaScript just use ~~Simi~~ similar name with Java! JavaScript combined with lots of programming language!

+2

7. "{}" is syntactic sugar for "new Object()" in JavaScript.

TRUE

+2

8. In JavaScript, functions that do not have any sort of explicitly defined return statement will return `undefined`.

TRUE.

+2

+0

9. In JavaScript, functions that do not have any sort of explicitly defined return statement will return ?

?

return undefined

10. In JavaScript, once you set the type of a variable, you are committed to that type.

+2

False! changed

```
var a = 4; type of Number
a = "4"; type of String
```

18

Multiple Choice: Circle the correct answer (16pts)

1. The correct syntax to implement inheritance in JavaScript is.
   a. `class newClass inherits oldClass`
   b. `class newClass extends oldClass`
   c. `class newClass : public oldClass`
   d. `newClass :: oldClass`

2. Which of these choices is another valid way to write the function `foo` defined in the box?

```
function foo(){
  return x + x;
}
```

   a. `let foo = function(){return x*x;}`
   b. `foo(x) = x+x;`
   c. `let foo = x => x+x;`
   d. `foo fighters(x){return x+x;}`

3. Being able to reference a specific instance of a local binding in an enclosing scope is called:
   a. Recursion
   b. Closure
   c. Nested Scope
   d. Destructuring

4. What does DOM stand for?
   a. Descriptive Object Model
   b. Desktop Oriented Methods
   c. Document Object Model
   d. Declarative Object Model

5. If you put a file or function in ___, JavaScript will throw errors in scenarios it would usually allow.
   a. Debug Mode
   b. Arbitrary Mode
   c. Strict Mode
   d. Test Mode

6. What programming paradigm is JavaScript?
   a. Imperative
   b. Functional
   c. Declarative
   d. All of the Above

7. In JavaScript, the result of "7" + 3 is 73. This is an example of:
   a. Type coercion
   b. Type conversion
   c. Type casting
   d. Type inference

8. Which statement best describes the purpose of the call stack?
   a. Keep track of program progress during execution.
   b. Perform the calculations a program requires.
   c. Store non-local objects a program creates dynamically.
   d. Insert any necessary dynamic checks of a program.

...it is true.

Qixiang Lin

Short-Medium Answer:

1. What was the animal Jacques turned into in the 4th chapter of *Eloquent JavaScript* (3rdEdition)? (1pt)

+0.75 Squirrel (wrong spelling) 松鼠 !!

2. Indicate the portion of code that is the "temporal dead zone" for variable x (2 pt)

```
let iscute = false;
y => {
  drive("shelter");
  walk("Max");
  let x = 20;
  adopt("Max");
  iscute= true;}
```

+2 3 ⇒ temporal dead zone

3. How does a compiler differ from an assembler? (2 pts)

+2 an assembler ~~translat is low-level~~ to translate assembly langnage to machine code! It is great flexibility but difficult to learn

a Compile traslate high-level langnage to assembly langnage or machine code!

4. Write a BNF definition for the context free grammar of strings of the following form: $a^n\ bc^n$. Valid examples include: abc, aabcc, aaabccc, aaaabcccc. Then create a parse tree that shows "aaabccc" is a valid string in your BNF. (5 pts)

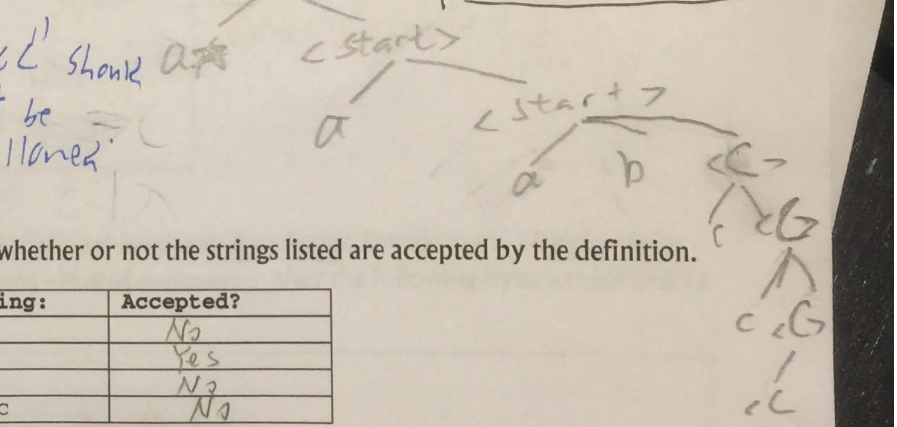
+2.5 start ::= aA | a start

A ::= bC

C ::= ε | cC | ∧

"abcC" should not be allowed

S ::= aSbC | abc



5. Examine the following BNF and answer whether or not the strings listed are accepted by the definition. (4 pts)

+4

```
<S> ::= "c" <X>+
<X> ::= "a" | "b"
```

String:	Accepted?
c	No
cab	Yes
bac	No
cabc	No

6. List (in order) the steps to compilation and explain how each step gets us closer to a compiled program. (10 pts)

+9 ① Scanning (lexical analysis) ~~from~~ translate character stream to token stream

② Parsing (syntax lexical) ~~to~~ combine token streams into parsing tree!

③ Semantics analysis and intermediate code generation, check parsing trees what is meaning of language. And intermediate code help compiler to translate to assembly language or target code!

(20.25)

④ Machine-Independent code is a optimization that make code better for manual

⑤ target code generation: produce a machine code for computer

⑥ machine-specific code generation is also optimization that make code effective

dynamic:

7. Give two examples of what is typically checked during static semantic analysis and two examples of a dynamic semantics check. (4 pts)

+1

① Divided by 0 ③ out of bound!

② Overflow

8. List 2 pros and 2 cons for using a compiled language vs an interpreted language (and a brief explanation as to why each reason is a pro / con if necessary for clarity): (8pts)

+3

portable ✓

Compiled		Interpreted	
Pros:	Cons:	Pros:	Cons:
1. ~~dev~~ - fast execution, fast runtime.	1. ✓ compiler need ✓ lost long time	1. user can modify interpreter!	1. ✗ No secrets
2. ✗ fewer runtime errors.	2. ✓ user cannot modify compile.	2. ~~check error~~ ✗ message easily	2. ✗ You cannot get optimization

Compiled translate high-level language to machine code.
Interpreted just translate and execute together!

9. What is the value of `message` after the following JavaScript code snippet executes? (3 pts)

+3

```
let mood = ":/";
function seeDog(){
  let mood = ":)";
  return name => name + "is now" + mood +"."
}
mood = "XD";
let message = seeDog()("James")
```

James is now :).

10. Here is a snippet from one of our favorite fictional languages DynaScript. DynaScript is a Pass By Value language with dynamic scoping. What is the value of `myanswers` after the following DynaScript runs? (5 pts)

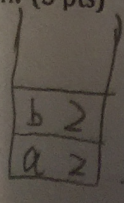
+5

```
let g = 2;
function a(){
  return g + x;
}
function b(){
  let x = 2;
  return a();
}
let x = 3;
let myanswers = [a(),b()];
```

[5, 4]

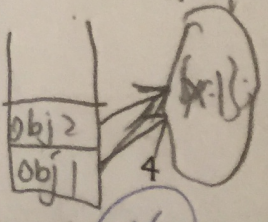
11. Your non-computer science-y friend is learning JavaScript because the language is taking over the world. She disagrees with you when you tell her that JavaScript is pass by value. Before you bring up the dangers of var out of nowhere, how do you explain that JavaScript actually is pass by value despite her impression? (3 pts)

+3



var a = 2;
var b = a;
create new location to give b;

var obj1 = {x:1};
var obj2 = obj1;
when obj2 = obj1;
obj2 copy obj1's memory address and point to obj1;



15

Procedural is imperative → what exact to do
↘ step by step

Imperative
Procedural

Qixiang Li

12. List 4 programming paradigms we have discussed in class and the distinguishing factors of each. (8 pts)

+6

Logic : rules and facts
Declaration : declare language! I have a cake! Declare what you want
Object-oriented : ~~create object~~ consist of many abstract objects, we can use them directly.
Function : this is mathematical calculation to calculate results

13. Why do some academics dislike the name "Programming Language Paradigms"? What would be a better name? (3 pts)

+1

Paradigm is a model or pattern.
~~Language paradigms~~ Language has more than one paradigms
Programming Paradigms

14. What characteristic of C/C++ did Distinguished Fellow of the British Computer Society, ACM Turing Award winner, and Fellow of the Royal Academy of Engineering, C.A.R. Hoare imply should be illegal? (He was also a Knight) (2 pts)

C/C++, C is imperative, C++ also combined C and has ~~a little~~ object-oriented!

Bounds check!

15. List at least 2 reasons different programming languages exist. (2 pts)

Programming language have different features for different people! such as object-oriented - functions

+1

Buisness and Personal Use!

second reason?

Deeper Thought:

1. Why do you suppose only variable declarations are hoisted and not the declaration and assignment? Hint: Functions can return values. (2 pts)

let x = 5;
let x = f();

function declaration also is hoisted, if assignment ~~is~~ also is hoisted, it will cause reference error!

if hoisted,

8

Qixiang Cai

Congratz! You're done! You should always go back through and check your work. If you've done that and don't want to leave yet or check your work a fourth or fifth time, here's some total-nonsense VALID JavaScript you can try to figure out for an extra point!

Extra Credit:

What does stranJS() return? (2pt) 1 point for correct answer, 1 point for correct justification.

```
function stranJS(){
    let x = 5;
    let b = new Boolean(false);
    if(b){
        if("false"){
          if(x=0){
              return x + false * 2;
          }else{
              return false + true + x;
          }

        }else{
            let y = 0/0; // whoops
            if(NaN == y){
                return x++;
            }else{
                return false * true + true/0.5 + x;
            }
        }
    }else{
        if("false"){
            if(x=0){
                return x || (true + false + true) / true;
            }else{
                return ++x * 10;
            }
        }else{
            let x = 0/0; //there I go again.
            if(NaN == x){
                return x++;
            }else{
                return x = 3;
            }
        }
    }
}
```

b is object ⇒ true.
if ("false") ⇒ true
x=0 ⇒ ~~false~~ ✓
return 5 + 0*2
= 5

= false + true + x
= 0 + 1 + 0
= 1.