7 9?3

1

NATIONAL UNIVERSITY OF SINGAPORE

CS1101S — PROGRAMMING METHODOLOGY

(AY2019/2020 SEMESTER 1)

READING ASSESSMENT 2, REDACTED EDITION OF 2020/21

Time Allowed: **40 Minutes**

INSTRUCTIONS

1. This question paper comprises **NINE (9)** printed pages, including this page.

2. You are also provided with **one OCR Form** to write your answers.

3. Clearly **write** and **shade** your **STUDENT NUMBER** in **SECTION B** of your **OCR Form** using a **2B PENCIL**.

4. You do not need to write in **SECTION A** of your **OCR Form**.

5. There are **17** multiple-choice questions. Each question has one correct answer. **1 mark** is awarded for each correct answer and there is no penalty for a wrong answer.

6. The full score is **17 marks**.

7. Answer **ALL** questions.

8. Use only a **2B PENCIL** to **shade** your answers on your **OCR Form**.

9. This is a **CLOSED BOOK** assessment, but you are allowed to bring in one A4 sheet of notes (handwritten or printed on both sides).

10. **Submit only the OCR Form**.

(1) What is the single-digit **number** at the **top-right corner** on the **front page** of this question paper? (**Important**: Please make sure your answer is correct because it determines how we mark your answers to all the subsequent questions.)

A. 1
B. 2
C. 3
D. 4
E. 5

(2) What is the order of growth of the *runtime* of the following function in terms of N using the Θ notation? Note that N is a non-negative integer value.

```
function funX(N) {
    let sum = 0;
    for (let x = 1; x <= N; x = x + 1) { sum = sum + 1; }
    for (let y = 2 * N; y >= 1; y = y - 1) { sum = sum + 1; }
    return sum;
}
```

A. $\Theta(1)$
B. $\Theta(N)$
C. $\Theta(N^2)$
D. $\Theta(N^3)$
E. None of the above

(3) What is the order of growth of the *runtime* of the following function in terms of N using the Θ notation? Note that N is a non-negative integer value.

```
function funY(N) {
    let sum = 0;
    for (let x = 1; x <= N; x = x + x) {
        for (let y = 1; y <= N; y = y + 2) {
            sum = sum + 1;
        }
    }
    return sum;
}
```

A. $\Theta(1)$
B. $\Theta(\log N)$
C. $\Theta(N)$
D. $\Theta(N \log N)$
E. $\Theta(N^2)$

```
for( i=0; i<2N; i = i+ N).
```

2

(4) What is the order of growth of the *runtime* of the following function in terms of N using the Θ notation? Note that N is a non-negative integer value.

```
function funZ(N) {
    let sum = 0;
    for (let i = 0; i <= 1024*N; i = i + (N / 2)) {
        sum = sum + 1;
    }
    return sum;
}
```

A. $\Theta(1)$
B. $\Theta(\log N)$
C. $\Theta(N)$
D. $\Theta(N \log N)$
E. $\Theta(N^2)$

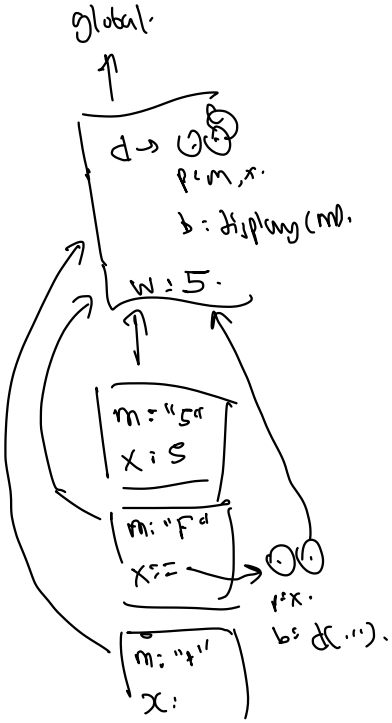
(5) The *runtime* of Program P has order of growth $\Theta(g(n))$. Which of the following is correct?

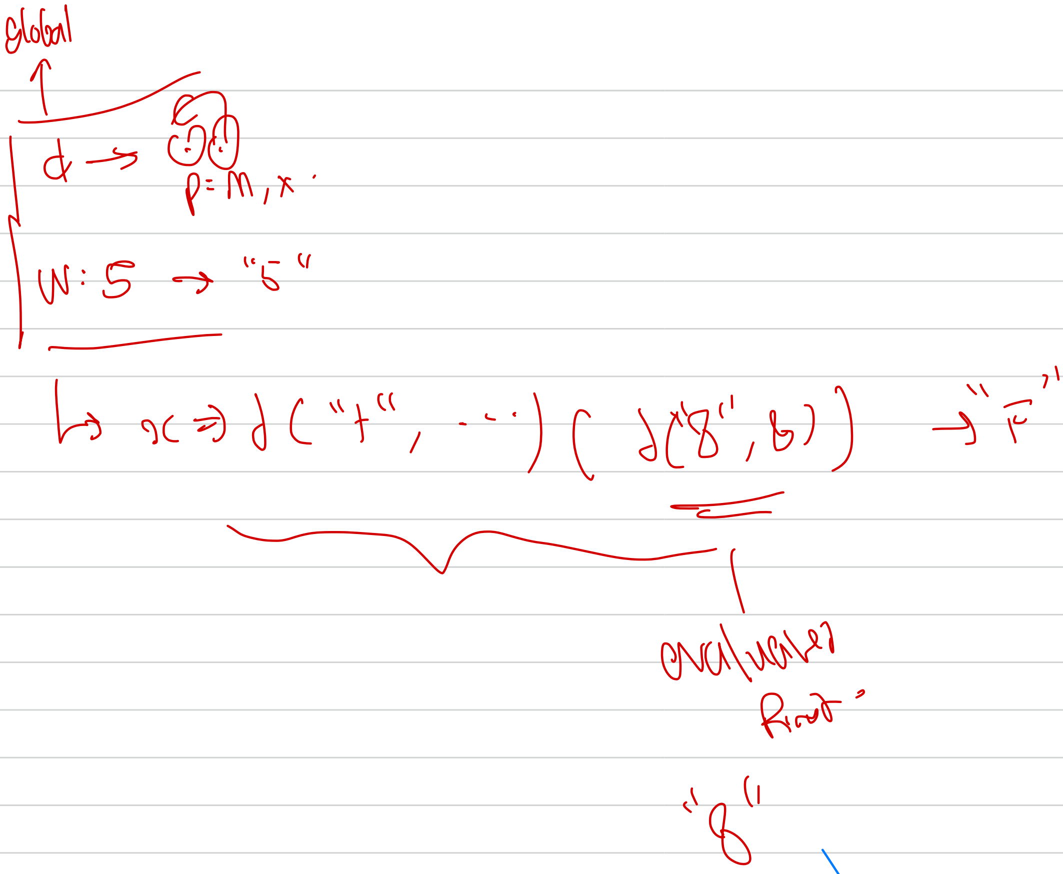
A. Its order of growth in *space* is $\Theta(g(n))$.
B. Its order of growth in *space* is $\Omega(g(n))$.
C. Its order of growth in *space* is $O(g(n))$.
D. Its order of growth in *space* is $\Theta(n)$.
E. None of the above.

(6) What is the sequence of values printed by the `display` function when the following program is evaluated?

```
function d(m, x) {
    display(m);
    return x;
}
const w = d("5", 5);
d("F", x => d("+", d("w", w) + d("x", x)) ) (d("8", 8));
// same as (x => w + x)(8)
```

A. "5" "F" "+" "w" "x" "8"
B. "5" "w" "x" "+" "F" "8"
C. "5" "F" "w" "x" "+" "8"
D. "5" "8" "w" "x" "+" "F"
E. "5" "F" "8" "w" "x" "+"

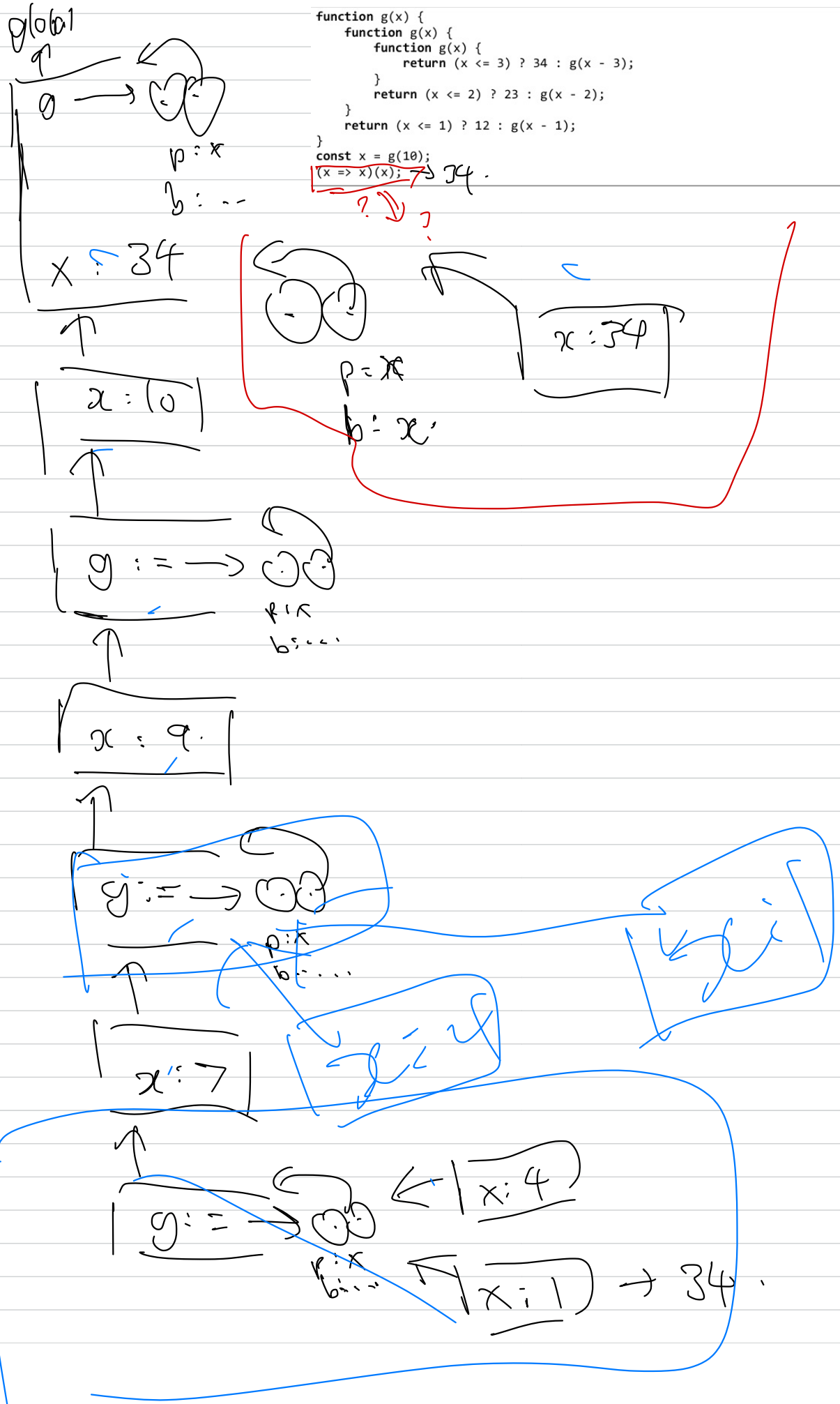




N
N²

```
arr = []
for(i=0; i<N; i++){
    char *a = malloc(N);
    arr[i*N] = 5;
}
```

```
function g(x) {
    function g(x) {
        function g(x) {
            return (x <= 3) ? 34 : g(x - 3);
        }
        return (x <= 2) ? 23 : g(x - 2);
    }
    return (x <= 1) ? 12 : g(x - 1);
}
const x = g(10);
(x => x)(x); → 34.
```



Consider the following Source program for the next 4 questions:

Program A:

```
function g(x) {
    function g(x) {
        function g(x) {
            return (x <= 3) ? 34 : g(x - 3);
        }
        return (x <= 2) ? 23 : g(x - 2);
    }
    return (x <= 1) ? 12 : g(x - 1);
}
const x = g(10);
(x => x)(x);
```

(7) During the evaluation of Program A, how many names does the program environment frame contain?

A. 0
B. 1
C. 2
D. 3
E. More than 3

(8) How many environment frames get created during the evaluation of Program A? (Do not count the global environment frame.)

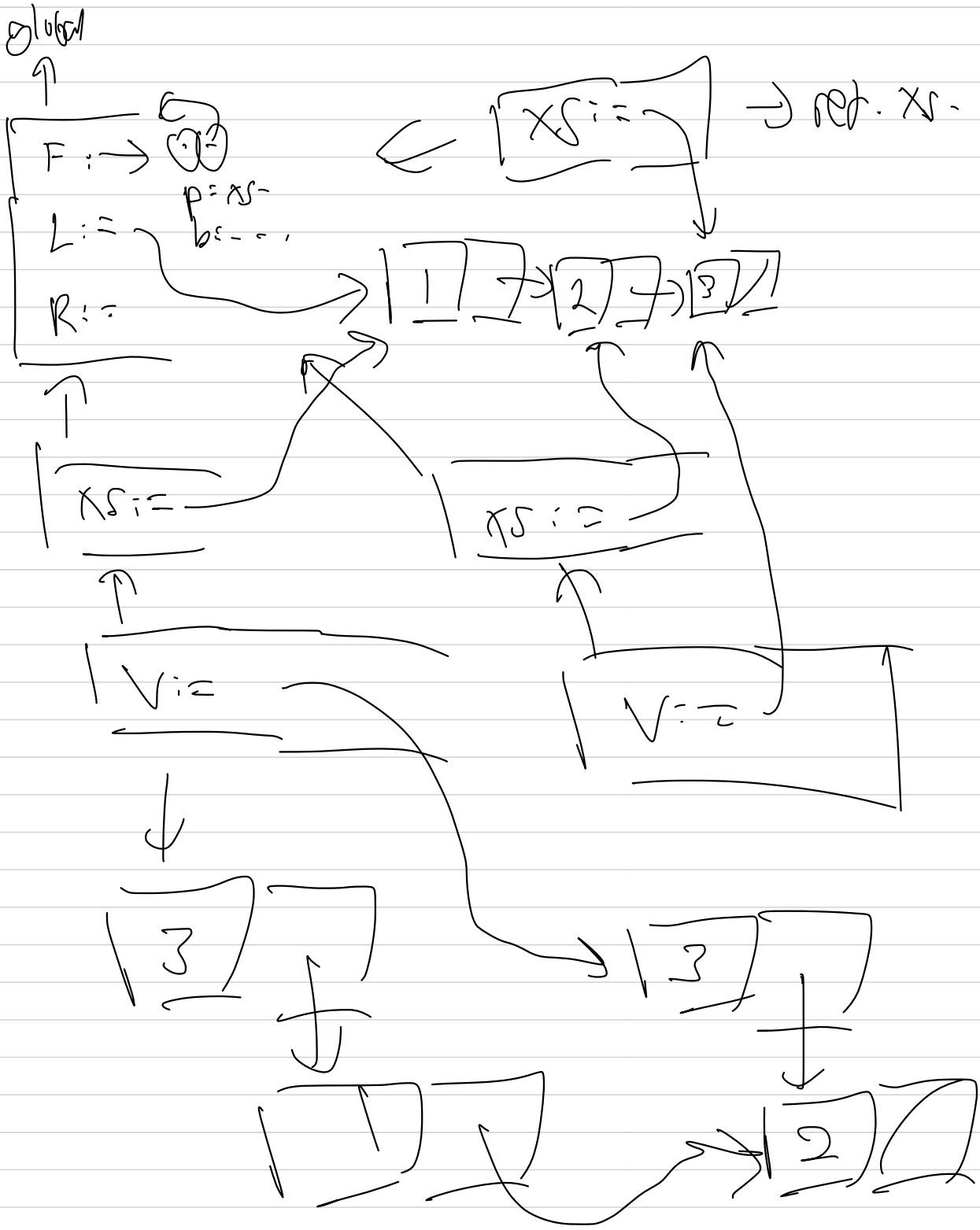
A. 9
B. 7
C. 6
D. 5
E. None of the above

(9) Of the environment frames that get created during the evaluation of Program A, how many extend the program environment *directly*?

A. 1
B. 2
C. 3
D. 4
E. More than 4

(10) How many function objects get created during the evaluation of Program A?

A. 1
B. 2
C. 3
D. 4
E. More than 4


global
F :=
p := xs
L :=
R :=
xs :=
ref. xs
1
2
3
xs :=
xs :=
V :=
V :=
3
3
1
2

Consider the following Source program for the next 3 questions:

Program B:

```
function F(xs) {
    if (is_null(tail(xs))) {
        return xs;
    } else {
        const v = F(tail(xs));
        return pair(head(v), pair(head(xs), tail(v)));
    }
}
const L = list(1, 2, 3);
const R = F(L);
R;
```

(11) How many environment frames get created during the evaluation of Program B? (Do not count the global environment frame. We assume that the application of a primitive function does not create any frame. See Appendix for a list of primitive functions.)

A. 6
B. 5
C. 4
D. 3
E. None of the above

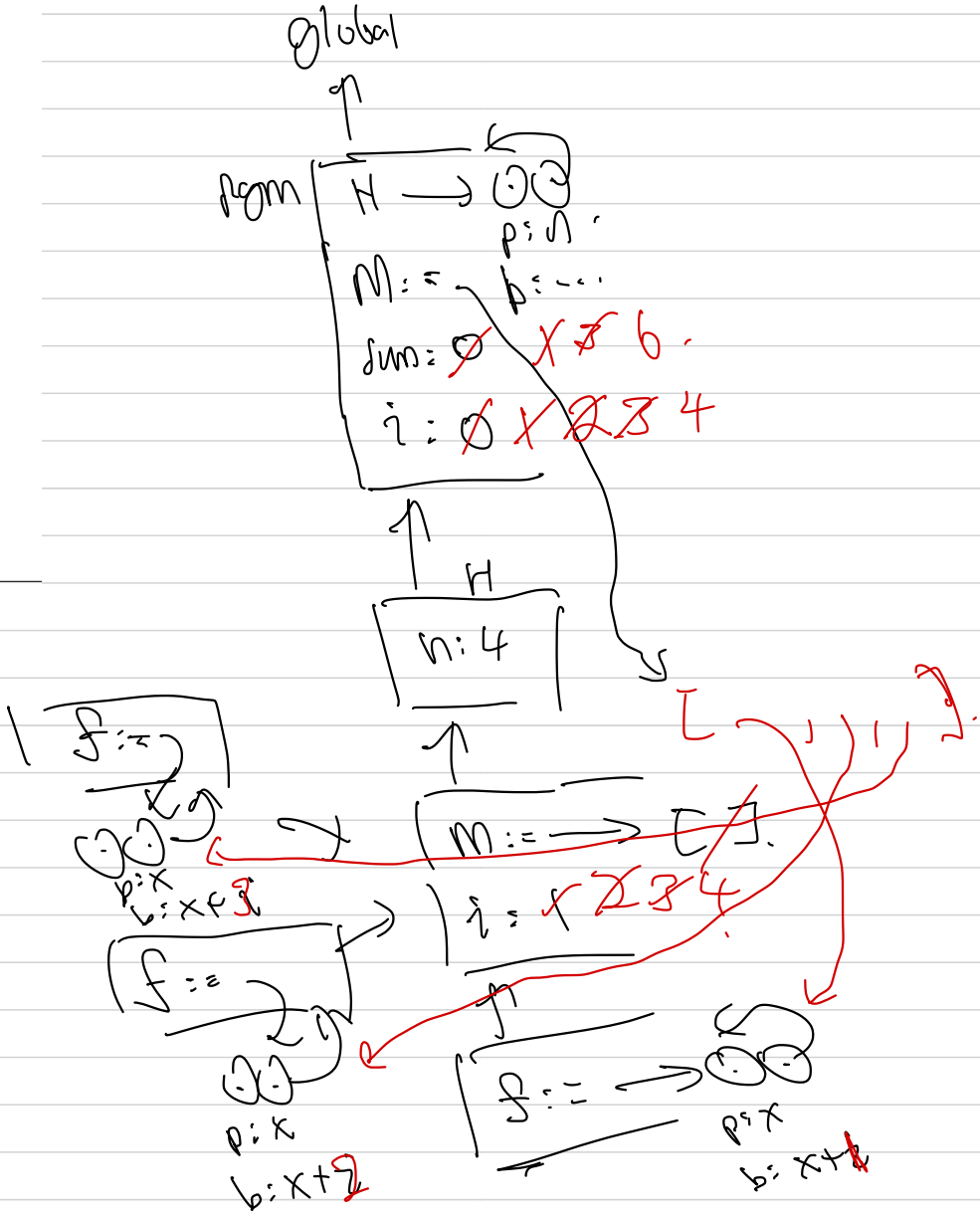
(12) Of the environment frames that get created during the evaluation of Program B, how many extend the program environment *directly*?

A. 5
B. 4
C. 3
D. 2
E. None of the above

(13) How many pairs get created during the evaluation of Program B?

A. 3
B. 5
C. 6
D. 7
E. None of the above

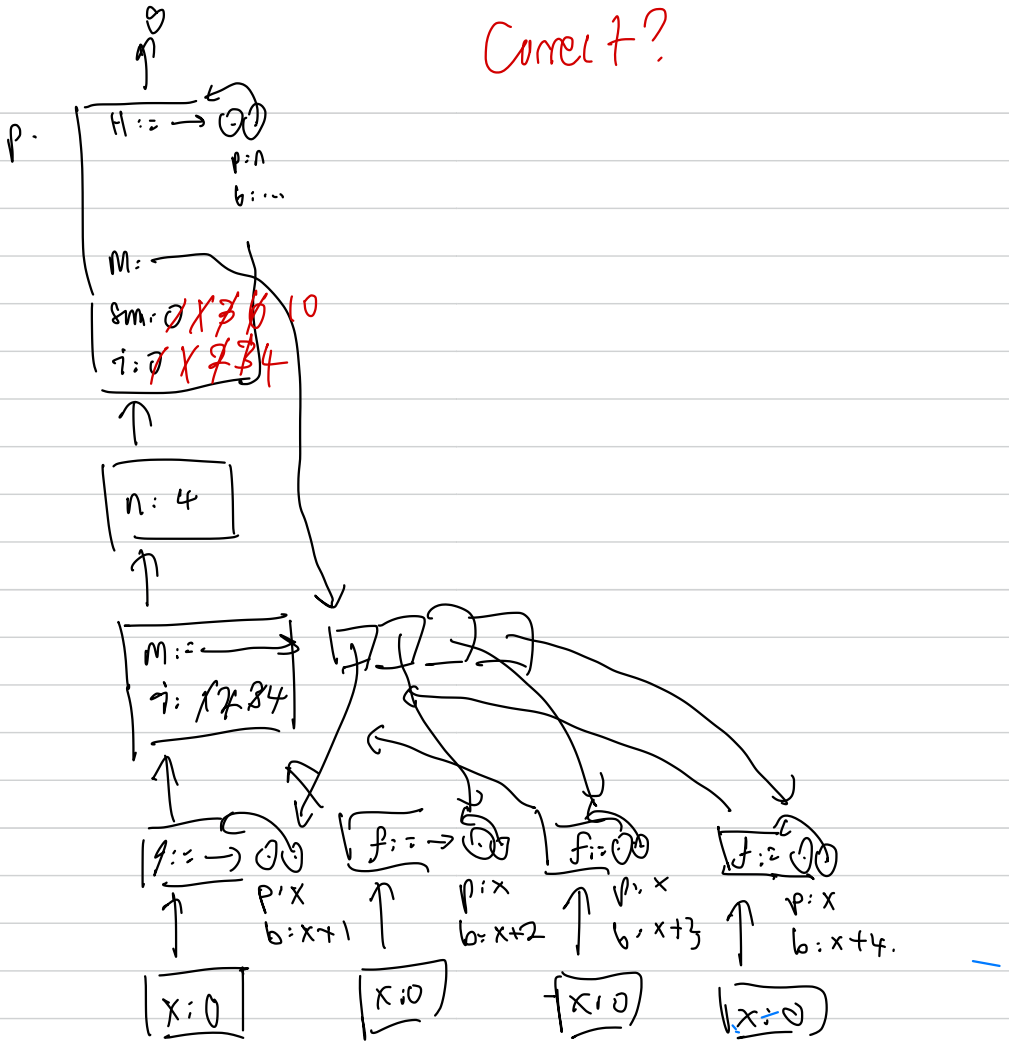
```
function H(n) {
    let M = [];
    let i = 1;
    while (i <= n) {
        const f = x => x + i;
        M[i - 1] = f;
        i = i + 1;
    }
    return M;
}
const M = H(4);
let sum = 0;
let i = 0;
while (i < 4) {
    sum = sum + M[i](0);
    i = i + 1;
}
sum;
```



```
function H(n) {
    let M = [];
    let i = 1;
    while (i <= n) {
        const f = x => x + i;
        M[i - 1] = f;
        i = i + 1;
    }
    return M;
}
const M = H(4);
let sum = 0;
let i = 0;
while (i < 4) {
    sum = sum + M[i](0);
    i = i + 1;
}
sum;
```

while this is satisfied

Correct?



Consider the following Source program for the next 4 questions:

Program C:

```
function H(n) {
    let M = [];
    let i = 1;
    while (i <= n) {
        const f = x => x + i;
        M[i - 1] = f;
        i = i + 1;
    }
    return M;
}
const M = H(4);
let sum = 0;
let i = 0;
while (i < 4) {
    sum = sum + M[i](0);
    i = i + 1;
}
sum;
```

(14) What is the value of `sum` at the end of the evaluation of Program C?

A. 6
B. 10
C. 15
D. 16
E. 20

(15) How many environment frames get created during the evaluation of Program C? (Do not count the global environment frame.)

A. 2
B. 3
C. 6
D. 10
E. None of the above

(16) Of the environment frames that get created during the evaluation of Program C, how many extend the program environment *directly*?

A. 1
B. 2
C. 5
D. 9
E. None of the above

[1]

(17) How many function objects get created during the evaluation of Program C?

A. 1
B. 4
C. 5
D. 9
E. None of the above

B. X C.

——— END OF QUESTIONS ———

Appendix

Primitive Functions

The following are some of the primitive functions in Source §3:

- `display(a)`
- `pair(x, y)`
- `is_pair(x)`
- `head(x)`
- `tail(x)`
- `is_null(xs)`
- `list(x1, x2,..., xn)`
- `set_head(p, x)`
- `set_tail(p, x)`
- `array_length(x)`

99+7.

20 min.

(blank page)

(blank page)