Name:	**Date:**
Vince Allen D. Cristal	February 4, 2026
School:	**Course:**
LSPU - San Pablo City Campus	On-The-Job Training

1. Inventory Tracker

Code:

```
let inventory = ["apple", "banana", "cherry", "date", "elderberry"];
let inventoryCount = [10, 15, 5, 20, 7];

for (let i = 0; i < inventory.length; i++) {
    console.log(`${inventory[i]}: ${inventoryCount[i]}`);
}
```

Output:

apple: 10

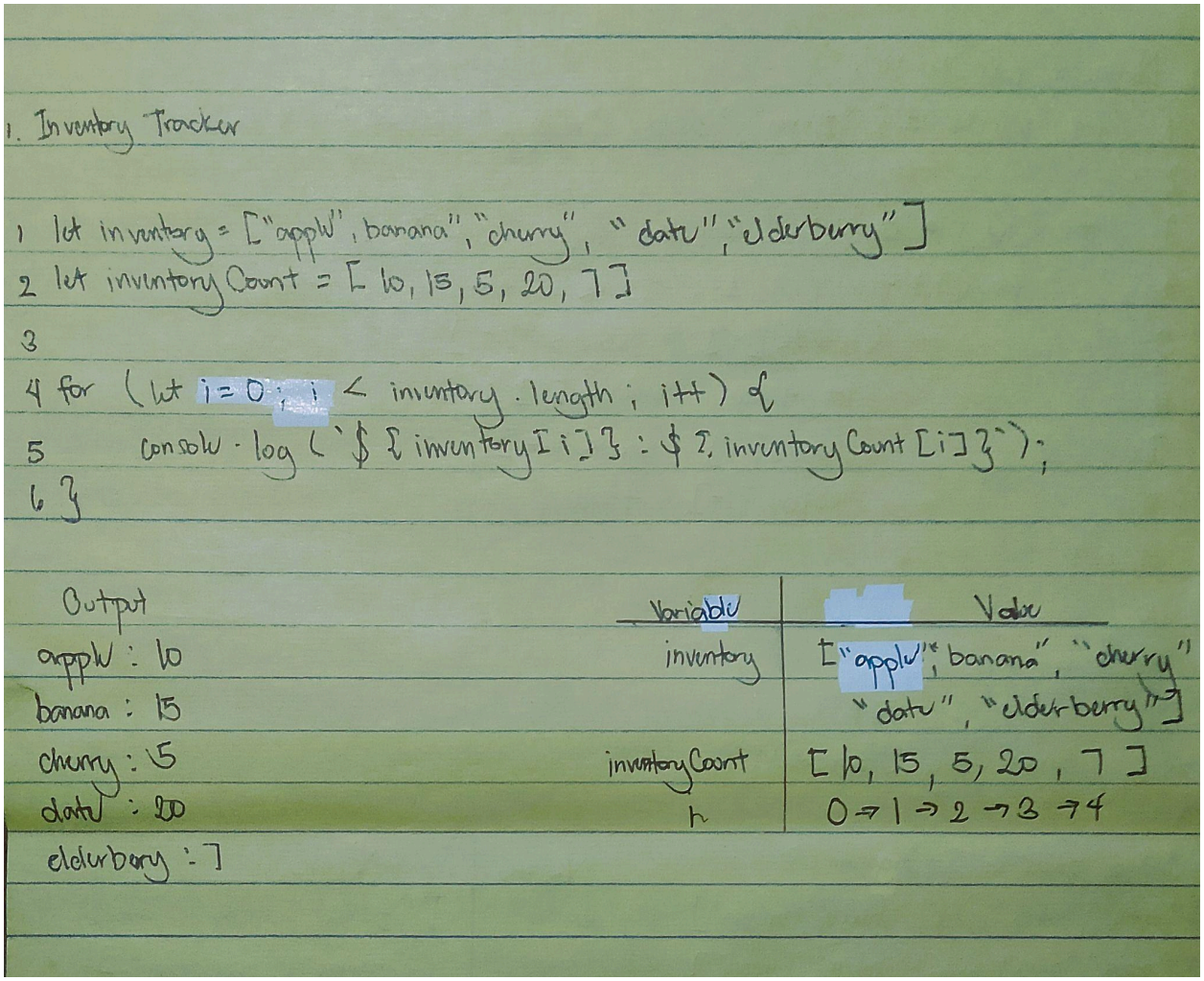
banana: 15

cherry: 5

date: 20

elderberry: 7

Photo of T-diagram:



1. Inventory Tracker

```
1 let inventory = ["apple", banana", "cherry", "date", "elderberry"]
2 let inventoryCount = [10, 15, 5, 20, 7]
3
4 for (let i=0; i < inventory.length; i++) {
5     console.log(`${inventory[i]}: ${inventoryCount[i]}`);
6 }
```

Output	Variable	Value
apple: 10	inventory	["apple", banana", "cherry" "date", "elderberry"]
banana: 15		
cherry: 5	inventoryCount	[10, 15, 5, 20, 7]
date: 20	i	0→1→2→3→4
elderberry: 7		

2. Alphabetical Sorting

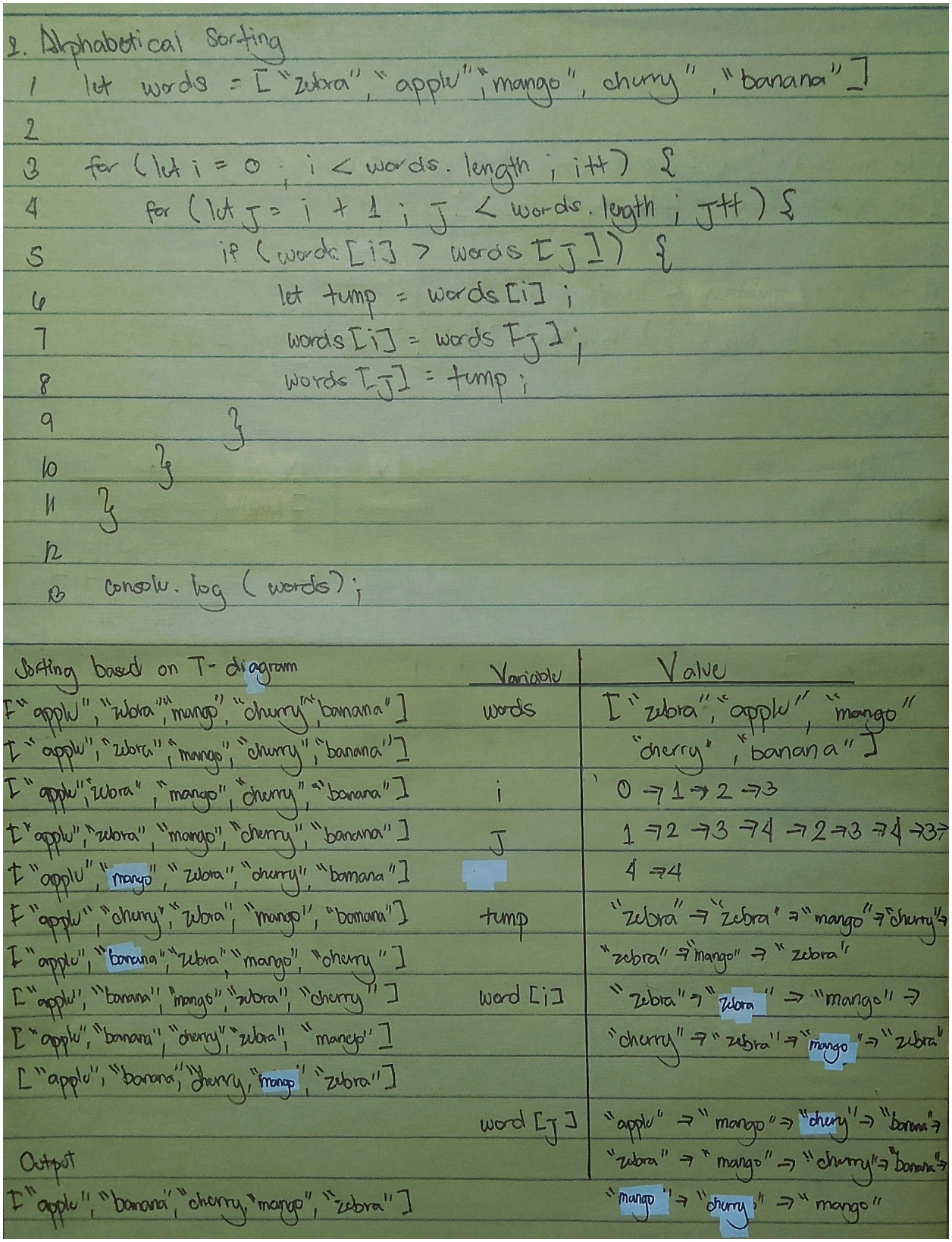
Code:

```
let words = ["zebra", "apple", "mango", "cherry", "banana"];

for (let i = 0; i < words.length; i++) {
  for (let j = i + 1; j < words.length; j++) {
    if (words[i] > words[j]) {
      let temp = words[i];
      words[i] = words[j];
      words[j] = temp;
    }
  }
}

console.log(words);
```

Output:

["apple", "banana", "cherry", "mango", "zebra"]

Photo of T-diagram:



1. Alphabetical Sorting

```
1   let words = ["zebra", "apple", "mango", "cherry", "banana"]
2
3   for (let i = 0 ; i < words.length ; i++) {
4       for (let j = i + 1 ; j < words.length ; j++) {
5           if (words[i] > words[j]) {
6               let temp = words[i] ;
7               words[i] = words[j] ;
8               words[j] = temp ;
9           }
10      }
11  }
12
13  console.log (words);
```

Sorting based on T-diagram

Variable	Value
words	["zebra", "apple", "mango" "cherry", "banana"]
i	0 → 1 → 2 → 3
j	1 → 2 → 3 → 4 → 2 → 3 → 4 → 3 → 4 → 4
temp	"zebra" → "zebra" → "mango" → "cherry" → "zebra" → "mango" → "zebra"
word[i]	"zebra" → "zebra" → "mango" → "cherry" → "zebra" → "mango" → "zebra"
word[j]	"apple" → "mango" → "cherry" → "banana" → "zebra" → "mango" → "cherry" → "banana" → "mango" → "cherry" → "mango"

Output

["apple", "banana", "cherry", "mango", "zebra"]

3. Unique Array Builder

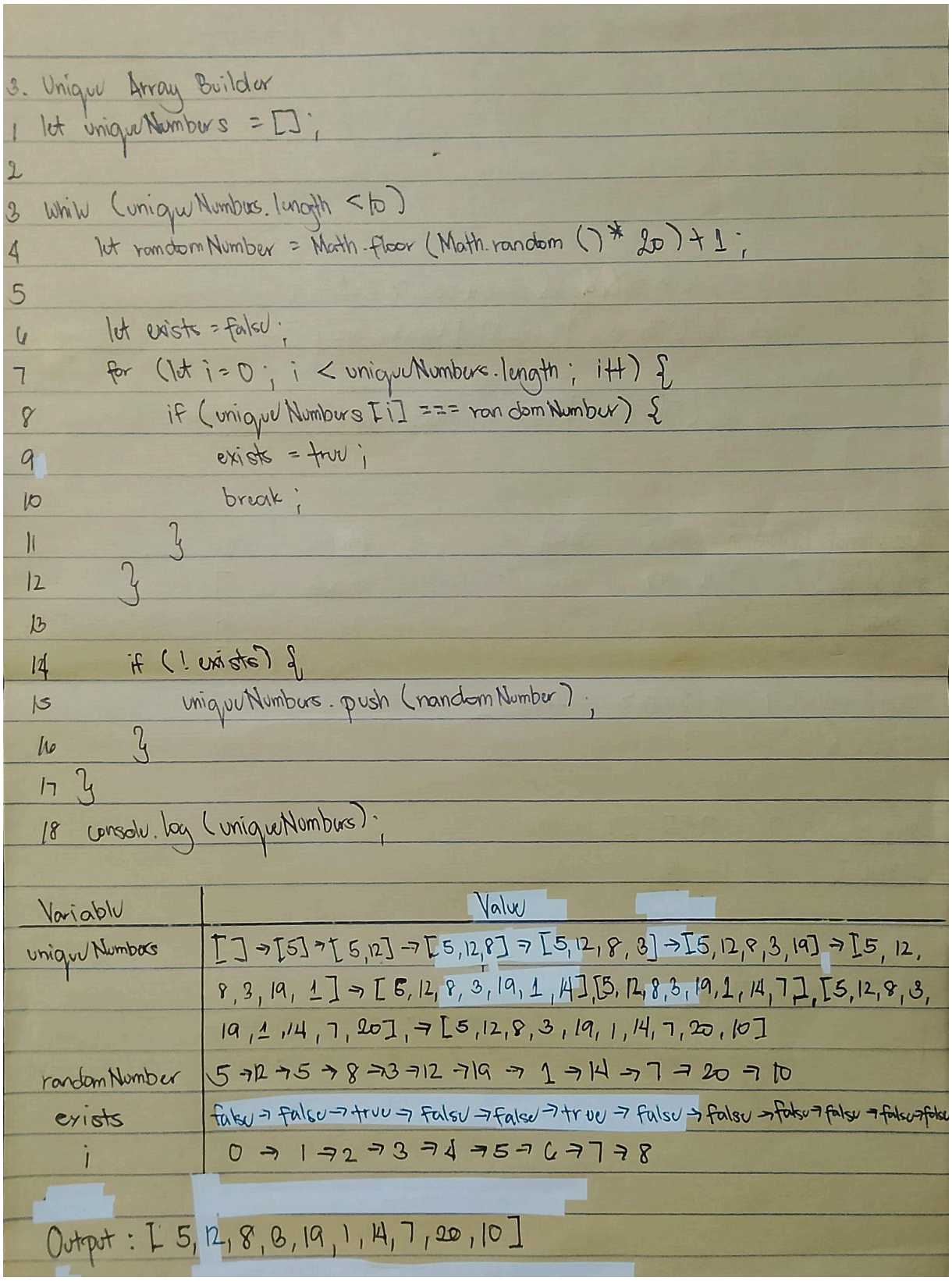
Code:

```
let uniqueNumbers = [];

while (uniqueNumbers.length < 10) {
  let randomNumber = Math.floor(Math.random() * 20) + 1;

  // Check if the randomNumber already exists in the array
  let exists = false;
  for (let i = 0; i < uniqueNumbers.length; i++) {
    if (uniqueNumbers[i] === randomNumber) {
      exists = true;
      break;
    }
  }

  // If it doesn't exist, add it to the array
  if (!exists) {
    uniqueNumbers.push(randomNumber);
  }
}

console.log(uniqueNumbers);
```

Output:

[5, 12, 8, 3, 19, 1, 14, 7, 20, 10]

Photo of T-diagram:



3. Unique Array Builder

```
1  let uniqueNumbers = [];
2
3  while (uniqueNumbers.length < 10)
4      let randomNumber = Math.floor(Math.random() * 20) + 1;
5
6      let exists = false;
7      for (let i = 0; i < uniqueNumbers.length; i++) {
8          if (uniqueNumbers[i] === randomNumber) {
9              exists = true;
10             break;
11         }
12     }
13
14     if (!exists) {
15         uniqueNumbers.push(randomNumber);
16     }
17 }
18 console.log(uniqueNumbers);
```

Variable	Value
uniqueNumbers	[] → [5] → [5,12] → [5,12,8] → [5,12,8,3] → [5,12,8,3,19] → [5, 12, 8, 3, 19, 1] → [5, 12, 8, 3, 19, 1, 14], [5, 12, 8, 3, 19, 1, 14, 7], [5, 12, 8, 3, 19, 1, 14, 7, 20], → [5, 12, 8, 3, 19, 1, 14, 7, 20, 10]
randomNumber	5 → 12 → 5 → 8 → 3 → 12 → 19 → 1 → 14 → 7 → 20 → 10
exists	false → false → true → false → false → true → false → false → false → false → false
i	0 → 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8

Output : [5, 12, 8, 3, 19, 1, 14, 7, 20, 10]

4. Triangle Checker

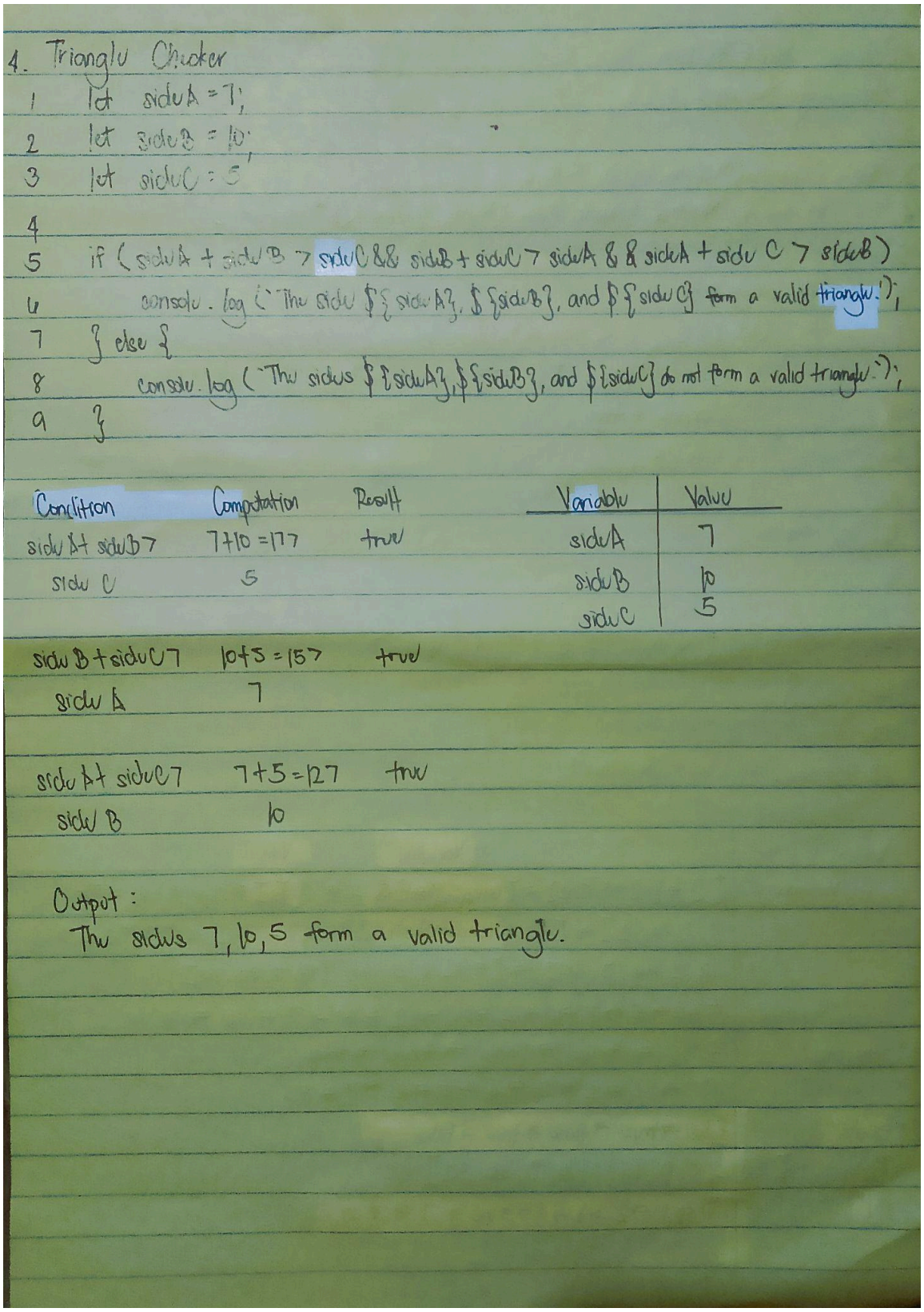
Code:

```
let sideA = 7;
let sideB = 10;
let sideC = 5;

if (sideA + sideB > sideC && sideB + sideC > sideA && sideA + sideC > sideB) {
  console.log(`The sides ${sideA}, ${sideB}, and ${sideC} form a valid triangle.`);
} else {
   console.log(`The sides ${sideA}, ${sideB}, and ${sideC} do not form a valid triangle.`);
}
```

Output:

The sides 7, 10, and 5 form a valid triangle.

Photo of T-diagram:



4. Triangle Checker

```
1   let sideA = 7;
2   let sideB = 10;
3   let sideC = 5;
4
5   if (sideA + sideB > sideC && sideB + sideC > sideA && sideA + sideC > sideB)
6       console.log(`The sides ${sideA}, ${sideB}, and ${sideC} form a valid triangle.`);
7   } else {
8       console.log(`The sides ${sideA}, ${sideB}, and ${sideC} do not form a valid triangle.`);
9   }
```

Condition	Computation	Result
sideA + sideB > sideC	7+10 = 17 > 5	true
sideB + sideC > sideA	10+5 = 15 > 7	true
sideA + sideC > sideB	7+5 = 12 > 10	true

Variable	Value
sideA	7
sideB	10
sideC	5

Output:
The sides 7, 10, 5 form a valid triangle.