Homework 4

● Graded

Student

Giancarlos Marte

Total Points

91 / 104 pts

Question 1

Inheritance

20 / 20 pts

✔ **– 0 pts** Correct

– 20 pts not answered

– 3 pts a) sort

– 3 pts a) print

– 1 pt a) not complete

– 6 pts a)

– 2 pts b) find the Stretchable shapes

– 2 pts b) stretch them by a factor of 2,

– 2 pts b) print

– 1 pt b) not complete

– 6 pts b)

– 4 pts c) print

– 8 pts c)

– 1 pt c) not complete

Question 2

Change Making

4 / 17 pts

- **– 0 pts** Correct
- **– 17 pts** not answered
- **– 15 pts** not answered
- **– 8 pts** a)
- **– 1.5 pts** a) greedy solution not complete
- ✔ **– 3 pts** a) greedy solution
- **– 2 pts** a) is it the best solution?
- ✔ **– 3 pts** a) If not the best solution, what is?
- ✔ **– 2 pts** b) not complete
- **– 3 pts** b) not complete
- ✔ **– 5 pts** b) not complete
- **– 4 pts** b) coinsUsed array
- **– 9 pts** b)

Question 3

Games: Tic Tac Toe

13 / 13 pts

- ✔ **– 0 pts** Correct
- **– 13 pts** Not Answered
- **– 5 pts** Tree not correct/missing
- **– 1 pt** Explanation not correct/ missing

Question 4

A Greedy Algorithm for Tic Tac Toe

9 / 9 pts

- ✔ **– 0 pts** Correct
- **– 9 pts** not answered
- **– 2 pts** Wong Percolation/explanation of the greedy method
- **– 5 pts** Wrong answer
- **– 2 pts** Incomplete percolation of the greedy method

Question 5

Nim

17 / 17 pts

✔ **– 0 pts** Correct

– 17 pts not answered

– 2 pts not complete

– 5 pts a) not complete

– 2 pts b) not complete

– 5 pts b) not complete

– 12 pts wrong

– 8 pts b)

Question 6

A Set Application

28 / 28 pts

✔ **– 0 pts** Correct

– 28 pts not answered

– 9 pts a not answered

– 10 pts b not answered

– 9 pts c not answered

– 2 pts Not optimal data structure

– 2 pts b/c not complete/correct

– 5 pts c incorrect

– 5 pts a incorrect

– 5 pts b incorrect

Question assigned to the following page: 1

1)

a.

```
Shape [ ] a = { new Circle( 2.0 ), new Rectangle( 1.0, 3.0 ), new Rectangle(4.0, 3.0)};

public static Shape[] arr sort(Shape[] arr) {
        Shape[] temp = arr;
        temp.sort((r1, r2) -> r1.compareTo(r2));
        for (Shape shape : temp) {
          System.out.println(shape.area());
        }
        return temp;
}
```

b.

```
public static void stretchShape(Shape[] arr) {
        Shape[] temp = arr;
        for (Shape s : temp) {
                if (s instanceof Stretchable) {
                        ((Stretchable) s).stretch(2);
                        System.out.println(s.area());
                }
        }
}
```

c.

```
TreeSet<Shape> treeShapeSet = new TreeSet<Shape>((r1, r2) -> r1.compareTo(r2));
for(Shape s : a) {
        treeShapeSet.add(s);
}
Iterator iterator = treeShapeSet.iterator();
while (iterator.hasNext()) {
        System.out.println(iterator.next.area());
}
```

Question assigned to the following page: 2

2)

a.

```
/* greedy algorithm solution */
int change = 60;
int quarter = 0;
int dime = 0;
int nickel = 0;
int penny = 0;
int totalCoins = 0;

while (change != 0) {
   if (change >= 25) {
         quarter++;
         change = change - 25;
   }
   else if (change >= 10) {
         dime++;
         change = change - 10;
   }
   else if (change >= 5) {
         nickel++;
         change = change - 5;
   }
   else {
         penny++;
         change = change - 1;
   }
}
totalCoins = quarter + dime + nickel + penny;
```

This is not the best solution because it sometimes doesn't work for a certain amount of change (like 30) and adding or taking away some coinage will affect it. The best solution is a dynamic programming implementation.

b.

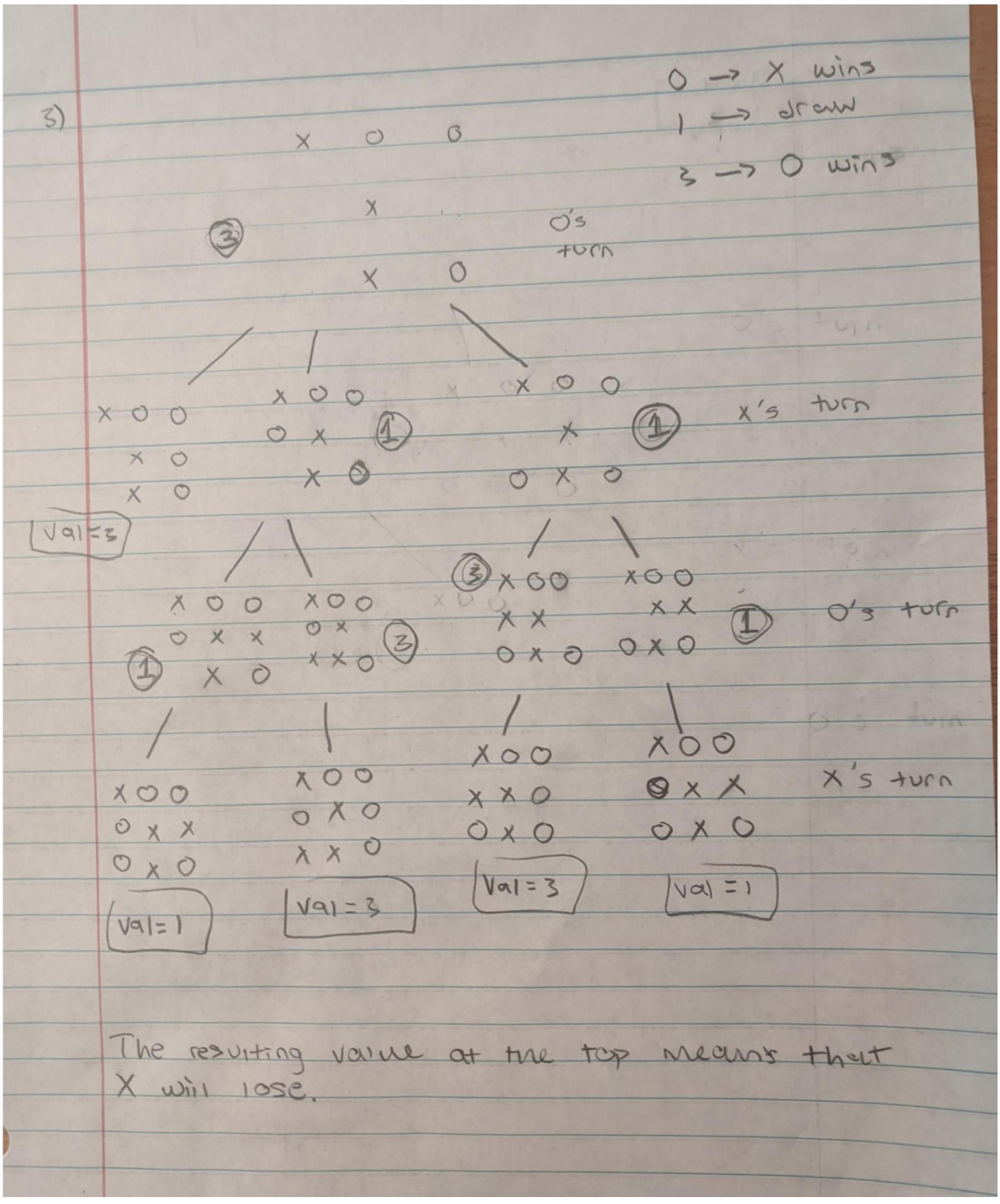
```
int[] cointsUsed = {1, 2, 3, 4, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 2, 3, 4, 5, 6, 2, 3, 4};
```

solution for change for 22 cents is 4

Question assigned to the following page: 3

3)

The resulting value at the top means that player x will lose.

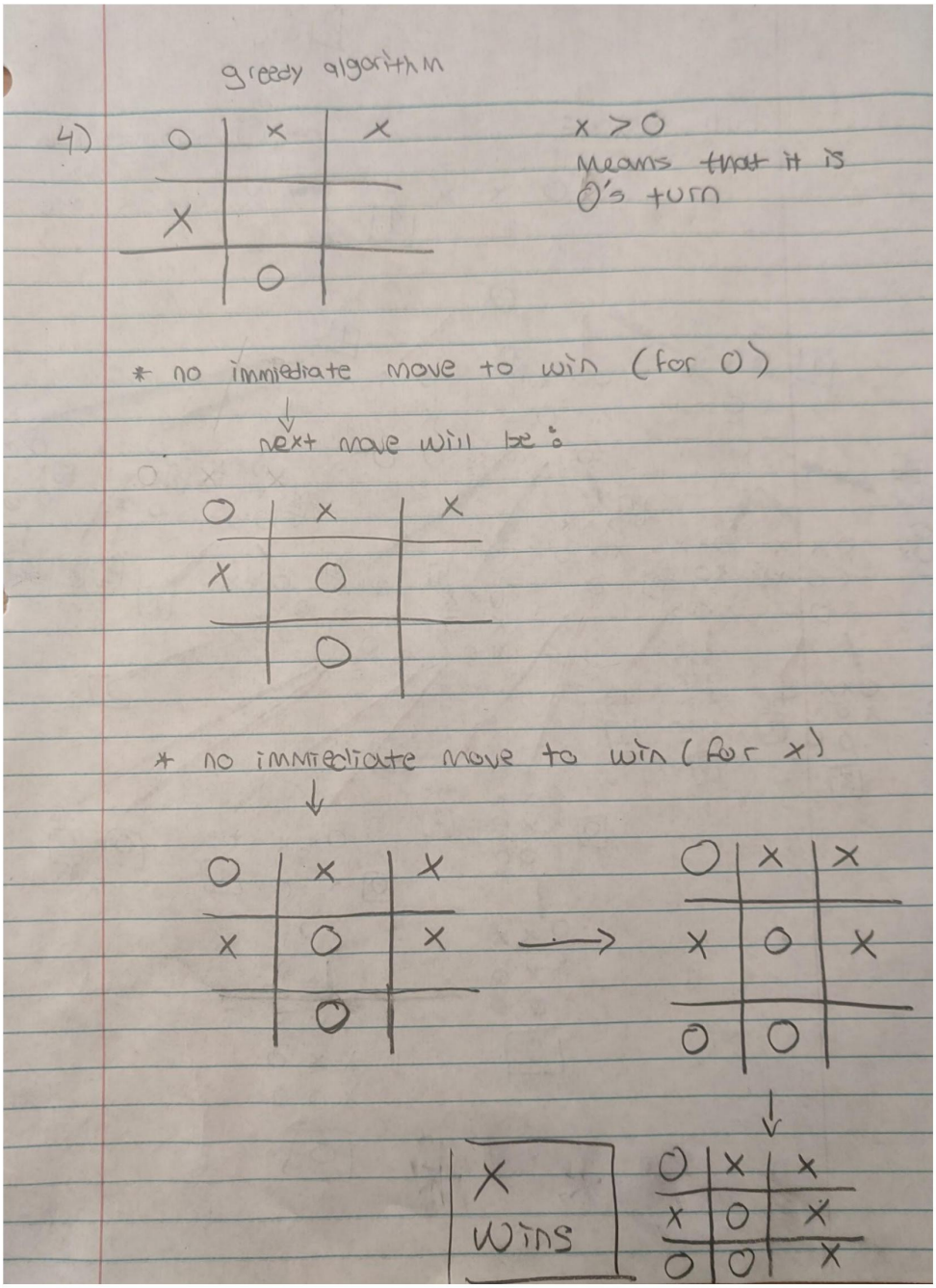


Question assigned to the following page: 4

4)

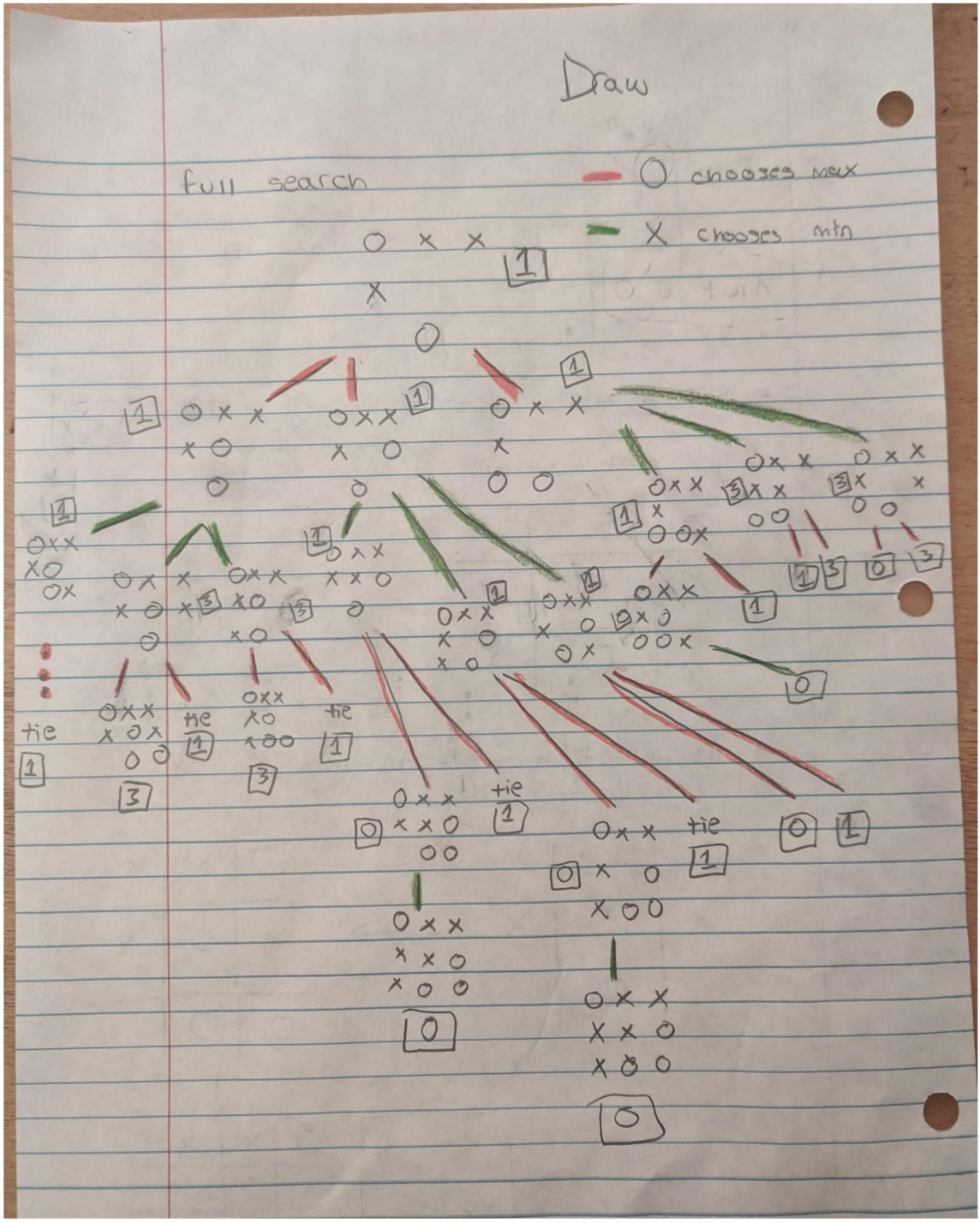
The greedy algorithm is not as good because X will always win, whereas in the full search the outcome will be a draw.

greedy



Question assigned to the following page: 4

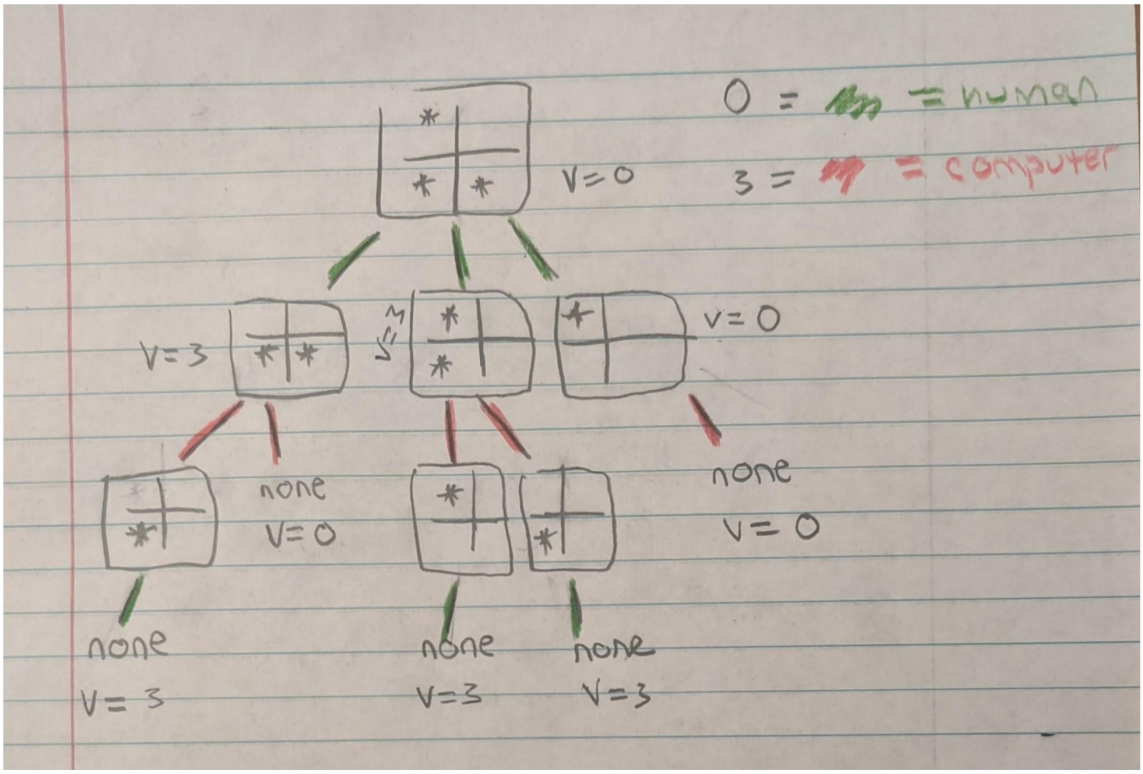
full search


Draw
full search
O chooses max
X chooses min
tie
tie
tie
tie
tie

Question assigned to the following page: 5

5)

The human's possibilities of winning are 100% if they choose 2 stars from the second row, other than that they will most likely lose.



Question assigned to the following page: 6

6)

a.

```
Set<Integer> clients = webIF.getClients();
// Set<Integer> prefs = webIF.getPrefs(id);

// client, prefs
TreeMap<Integer, Set<Integer> data = new TreeMap<>();

// load data
for (Integer i : clients) {
        data.put(i, webIF.getPrefs(i));
}
```

b.

```
public Integer findBestMatch(Integer id) {
        Set<Integer> intersects;
        Integer maxClient;
        Set<Integer> maxPrefs = new HashSet<>();
        Set<Integer> p = data.get(id);
        for (Integer i : clients) {
                intersects = new HashSet<>(p);
                intersects.retainAll(data.get(i));
                if (intersects.size() > maxPrefs.size()) {
                        max = i;
                        maxPrefs = intersects;
                }
        }
        return maxClient;
}
```

Question assigned to the following page: 6

c.

If someone turned in an empty questionnaire they will only be matched with others who have empty questionnaires (they will almost never have a match). If someone checked every box, they will have the option to be matched with almost anyone (they will always have a match). It is not true if A is the best match for B then vice versa will be true. This is because each might have more or less preferences than one another.

A better algorithm would be one that would have preferences that have priorities. If someone turns in an empty questionnaire they will be matched to someone at random. If someone checked every box, the highest priorities will be the most significant in matchmaking.