Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни
«Проектування алгоритмів»

“Пошук в умовах протидії, ігри з повною інформацією, ігри з елементом випадковості, ігри з неповною інформацією”

Виконав(ла) ІП-12 Волков Вадим Всеволодович 1.01.2023
(шифр, прізвище, ім’я, по батькові)

Перевірив Сопов Олексій Олександрович 1.01.2023
(прізвище, ім’я, по батькові)

Київ 2022

ЗМІСТ

1. МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Мета роботи - вивчити основні підходи до формалізації алгоритмів знаходження рішень задач в умовах протидії. Ознайомитися з підходами до програмування алгоритмів штучного інтелекту в іграх з повною інформацією, іграх з елементами випадковості та в іграх з неповною інформацією.

2. ЗАВДАННЯ

Для ігор з повної інформацією, згідно варіанту (таблиця 2.1) реалізувати візуальний ігровий додаток для гри користувача з комп'ютерним опонентом. Для реалізації стратегії гри комп'ютерного опонента використовувати алгоритм альфа-бета-відсікань. Реалізувати три рівні складності (легкий, середній, складний).

Для ігор з елементами випадковості, згідно варіанту (таблиця 2.1) реалізувати візуальний ігровий додаток, з користувацьким інтерфейсом, не консольним, для гри користувача з комп'ютерним опонентом. Для реалізації стратегії гри комп'ютерного опонента використовувати алгоритм мінімакс.

Для карткових ігор, згідно варіанту (таблиця 2.1), реалізувати візуальний ігровий додаток, з користувацьким інтерфейсом, не консольним, для гри користувача з комп'ютерним опонентом. Потрібно реалізувати стратегію комп'ютерного опонента, і звести гру до гри з повною інформацією (див. Лекцію), далі реалізувати стратегію гри комп'ютерного опонента за допомогою алгоритму мінімаксу або альфа-бета-відсікань.

Реалізувати анімацію процесу жеребкування (+1 бал) або реалізувати анімацію ігрових процесів (роздачі карт, анімацію ходів тощо) (+1 бал).

Реалізувати варто тільки одне з бонусних завдань.

Зробити узагальнений висновок лабораторної роботи.

Таблиця 2.1 – Варіанти

№	**Варіант**	**Тип гри**
1	Яцзи https://game-wiki.guru/published/igryi/yaczzyi.html	З елементами випадковості
2	Лудо http://www.iggamecenter.com/info/ru/ludo.html	З елементами випадковості
3	Генерал http://www.rules.net.ru/kost.php?id=7	З елементами випадковості
4	Нейтріко http://www.iggamecenter.com/info/ru/neutreeko.html	З повною інформацією
5	Тринадцять http://www.rules.net.ru/kost.php?id=16	З елементами випадковості
6	Индійські кості http://www.rules.net.ru/kost.php?id=9	З елементами випадковості
7	Dots and Boxes https://ru.wikipedia.org/wiki/Палочки_(игра)	З повною інформацією
8	Двадцять одне http://gamerules.ru/igry-v-kosti-part8#dvadtsat-odno	З елементами випадковості
9	Тіко http://www.iggamecenter.com/info/ru/teeko.html	З повною інформацією

10	Клоббер http://www.iggamecenter.com/info/ru/clobber.html	З повною інформацією
11	101 https://www.durbetsel.ru/2_101.htm	Карткові ігри
12	Hackenbush http://www.papg.com/show?1TMP	З повною інформацією
13	Табу https://www.durbetsel.ru/2_taboo.htm	Карткові ігри
14	Заєць і Вовки (за Зайця) http://www.iggamecenter.com/info/ru/foxh.html	З повною інформацією
15	Свої козирі https://www.durbetsel.ru/2_svoi-koziri.htm	Карткові ігри
16	Війна з ботами https://www.durbetsel.ru/2_voina_s_botami.htm	Карткові ігри
17	Domineering 8x8 http://www.papg.com/show?1TX6	З повною інформацією
18	Останній гравець https://www.durbetsel.ru/2_posledny_igrok.htm	Карткові ігри
19	Заєць и Вовки (за Вовків) http://www.iggamecenter.com/info/ru/foxh.html	З повною інформацією
20	Богач https://www.durbetsel.ru/2_bogach.htm	Карткові ігри
21	Редуду https://www.durbetsel.ru/2_redudu.htm	Карткові ігри
22	Эльферн https://www.durbetsel.ru/2_elfern.htm	Карткові ігри
23	Ремінь https://www.durbetsel.ru/2_remen.htm	Карткові ігри
24	Реверсі https://ru.wikipedia.org/wiki/Реверси	З повною інформацією
25	Вари http://www.iggamecenter.com/info/ru/oware.html	З повною інформацією
26	Яцзи https://game-wiki.guru/published/igryi/yaczzyi.html	З елементами випадковості
27	Лудо http://www.iggamecenter.com/info/ru/ludo.html	З елементами випадковості
28	Генерал http://www.rules.net.ru/kost.php?id=7	З елементами випадковості
29	Сим https://ru.wikipedia.org/wiki/Сим_(игра)	З повною інформацією
30	Col http://www.papg.com/show?2XLY	З повною інформацією
31	Snort http://www.papg.com/show?2XM1	З повною інформацією
32	Chomp http://www.papg.com/show?3AEA	З повною інформацією
33	Gale http://www.papg.com/show?1TPI	З повною інформацією
34	3D Noughts and Crosses 4 x 4 x 4 http://www.papg.com/show?1TND	З повною інформацією
35	Snakes http://www.papg.com/show?3AE4	З повною інформацією

3. ВИКОНАННЯ

1. Програмна реалізація алгоритму

1. Вихідний код

index.html

```
<!doctype html>
<html lang="en">
	<head>
		<title>21</title>
		<meta charset="UTF-8">
		<link rel="stylesheet" href="style.css">
		<script src="main.js" defer></script>
		<script src="anim.js" defer></script>
	</head>
	<body>
		<div class="center">
		<h1>Двадцять одно</h1>
		<div id="aiSelect">
			<h3>Оберіть складність</h3>
			<button id="ai4">Проста (занадто обережний)</button><br>
			<button id="ai3">Середня (занадто обережний)</button><br>
			<button id="ai2">Складна</button><br>
			<button id="ai1">Середня (занадто ризиковий)</button><br>
			<button id="ai0">Проста (занадто ризиковий)</button>
		</div>
		<div id="game" class="hidden">
			<table id="table">
				<tr>
					<th>Гравець</th>
					<th>Сума чисел</th>
					<th>Фішок</th>
				</tr>
				<tr>
					<td>Ви</td>
					<td>0</td>
					<td>10</td>
				</tr>
				<tr>
					<td>Комп'ютер 1</td>
					<td>0</td>
					<td>10</td>
				</tr>
			</table>
			<span id="pile" class="bold"></span>
			<br>
			<button id="throw" disabled>Кинути</button>
			<button id="next" disabled>Далі</button>
			<br>
			<canvas id="canvas" width="400" height="400"></canvas>
		</div>
		</div>
	</body>
</html>
```

main.js

```
const id = id => document.getElementById(id);
const create = type => document.createElement(type);

let players = [];
let startingPlayer = 0;
let currentPlayer = 0;
let humanFinished = null;
let pile = 0;
let difficulty = 0;
let round = 0;

id("ai0").addEventListener("click", selectDifficulty);
id("ai1").addEventListener("click", selectDifficulty);
id("ai2").addEventListener("click", selectDifficulty);
id("ai3").addEventListener("click", selectDifficulty);
id("ai4").addEventListener("click", selectDifficulty);
id("throw").addEventListener("click", async () => {
	await throwCube();
```

```
        if(players[currentPlayer].numberSum > 21) {
                humanFinished();
        }
});
id("next").addEventListener("click", () => {
        humanFinished();
});

function selectDifficulty(evt) {
        difficulty = [0.1,0.3,0.5,0.7,0.9][+evt.target.id[2]];

        players = [];
        players.push(new HumanPlayer());
        players.push(new AIPlayer());
        render();

        id("aiSelect").classList.add("hidden");
        id("game").classList.remove("hidden");

        play();
}

async function play() {
        startingPlayer = 0;
        round = 1;
        while(players.length > 1) {
                await playOneRound();
                await wait(2);
                startingPlayer = (startingPlayer+1) % players.length;
                round++;
                render();
        }
        alert(player.isAI ? "Комп'ютер переміг" : "Ви перемогли");
}

async function playOneRound() {

        // Забрати по одній фішці у купу
        for(let i=0; i<players.length; i++) {
                players[i].numberSum = 0;
                players[i].chipCount--;
                pile++;
        }

        // Дати кожному гравцю покидати кість
        currentPlayer = startingPlayer;
        render();
        let maxPlayers = [];
        let maxScore = 0;
        for(let i=0; i<players.length; i++) {
                let player = players[currentPlayer];

                await player.doTurns(maxScore, i);

                if(player.numberSum > maxScore && player.numberSum < 22) {
                        maxScore = player.numberSum;
                        maxPlayers = [];
                }
                if(player.numberSum == maxScore) {
                        maxPlayers.push(player);
                }
                await wait(1);
                currentPlayer = (currentPlayer+1) % players.length;
                render();
        }

        // Видати фішки переможцям
        while(pile >= maxPlayers.length) {
                for(let i=0; i<maxPlayers.length; i++) {
                        maxPlayers[i].chipCount++;
                        pile--;
                }
                render();
                await wait(0.5);
        }

        // Видалити гравців без фішок
        for(let i=0; i<players.length; i++) {
                if(players[i].chipCount == 0) {
                        players.splice(i, 1);
```

```
                        i--;
                }
        }

        render();
}


class Player {
        isAI;
        numberSum = 0;
        chipCount = 2;
}

class HumanPlayer extends Player {
        constructor() {
                super();
                this.isAI = false;
        }
        async doTurns() {
                id("throw").disabled = false;
                id("next").disabled = false;
                console.log("Human Turn")

                await new Promise(resolve => humanFinished = resolve); //when function given in
resolve is called, it makes await continue

                id("throw").disabled = true;
                id("next").disabled = true;
                console.log("Human Turn End", players[currentPlayer].numberSum);
        }
}

class AIPlayer extends Player {
        constructor() {
                super();
                this.isAI = true;
        }
        async doTurns(maxScoreSoFar, playerId) {
                const playersAfterMe = players.length - (playerId+1);
                const me = players[currentPlayer];
                console.log("AI Turn")
                while(me.numberSum < 16) {
                        await throwCube();
                        await wait(0.5);
                }

                const ifIGetThisMyChanceIs = {};
                for(let number=16; number<22; number++) {
                        let total = 0;
                        for(let i=16; i<22; i++) {
                                total += chanceToWinIfOtherThorws(number, i).chance;
                        }
                        ifIGetThisMyChanceIs[number] = total / 6;
                }

                let chanceIfIContinue = chanceToWinIfIThorwAgain(maxScoreSoFar,
players[currentPlayer].numberSum, playersAfterMe, ifIGetThisMyChanceIs);
                let chanceIfIFinish;
                if(playersAfterMe > 0) {
                        chanceIfIFinish = ifIGetThisMyChanceIs[players[currentPlayer].numberSum];
                } else {
                        chanceIfIFinish = players[currentPlayer].numberSum >= maxScoreSoFar ? 1 : 0;
                }
                console.log("При ",players[currentPlayer].numberSum,"якщо кинути, шанс",
chanceIfIContinue);
                console.log("При ",players[currentPlayer].numberSum,"якщо не кинути, шанс",
chanceIfIFinish);
                while(chanceIfIContinue > difficulty || (chanceIfIContinue > 0 && chanceIfIFinish ==
0)) {
                        await throwCube();
                        await wait(0.5);
                        chanceIfIContinue = chanceToWinIfIThorwAgain(maxScoreSoFar,
players[currentPlayer].numberSum, playersAfterMe, ifIGetThisMyChanceIs);
                        if(playersAfterMe > 0) {
                                chanceIfIFinish =
ifIGetThisMyChanceIs[players[currentPlayer].numberSum];
                        } else {
                                chanceIfIFinish = players[currentPlayer].numberSum >= maxScoreSoFar ?
1 : 0;
```

```
                        }
                        console.log("При ",players[currentPlayer].numberSum,"якщо кинути, шанс",
chanceIfIContinue);
                        console.log("При ",players[currentPlayer].numberSum,"якщо не кинути, шанс",
chanceIfIFinish);
                }
                console.log("AI Turn End", players[currentPlayer].numberSum);
        }
}


function chanceToWinIfIThorwAgain(maxScoreSoFar, myScore, playerAfterMe, ifIGetThisMyChanceIs,
depth="", history=null) {
        if(!history) history=myScore;
        let total = 0;
        for(let number=1; number<7; number++) {
                let chance = 0;
                if(myScore + number > 21) {
                        chance = 0;
                } else {
                        let chanceIfIContinue = chanceToWinIfIThorwAgain(maxScoreSoFar, myScore +
number, playerAfterMe, ifIGetThisMyChanceIs, depth+" ", history+"+"+number);
                        let chanceIfIFinish;
                        if(playerAfterMe > 0) {
                                chanceIfIFinish = ifIGetThisMyChanceIs[myScore + number];
                        } else {
                                chanceIfIFinish = (myScore + number) >= maxScoreSoFar ? 1 : 0;
                        }
                        chance = Math.max(chanceIfIContinue, chanceIfIFinish);
                }
                total += chance;
        }
        return total / 6;
}

function chanceToWinIfOtherThorws(thinkerScore, myScore) {
        let total = 0;
        for(let number=1; number<7; number++) {
                let chance = 0;
                if(myScore + number > 21) {
                        chance = 1; // Цей програв, перший виграв
                } else {
                        chance += chanceToWinIfOtherThorws(thinkerScore, myScore + number).chance;
                }
                total += chance;
        }
        let chanceIfIContinue = total / 6;
        let chanceIfIFinish;
        if(myScore > thinkerScore) {
                chanceIfIFinish = 0; // Набрав достатньо щоб перший програв
        } else {
                chanceIfIFinish = 1; // Не набрав достатньо щоб перший програв
        }
        chance = Math.min(chanceIfIContinue, chanceIfIFinish);
        return {chance, chanceIfIContinue, chanceIfIFinish};
}

function render() {
        const table = id("table");
        const tbody = table.children[0];
        const elements = tbody.children;
        while(elements.length > 1) {
                elements[1].remove();
        }
        let aiId = 1;
        for(let i=0; i<players.length; i++) {
                const player = players[i];

                const name = create("td");
                name.innerText = (currentPlayer == i ? "> ":"") + (player.isAI ? "Комп'ютер "+(aiId+
+) : "Ви");

                const sum = create("td");
                sum.innerText = player.numberSum;
                if(player.numberSum > 21) sum.classList.add("red");

                const chips = create("td");
                chips.innerText = player.chipCount;
                if(player.chipCount < 1) chips.classList.add("red");
```

```
                const tr = create("tr");
                tr.append(name, sum, chips);
                tbody.append(tr);
        }
        id("pile").innerText = "Купа: "+pile+" Раунд: "+round;
}

async function wait(time) {
        return new Promise(resolve => setTimeout(resolve, time*1000));
}

// Ця фунція перевизначається
async function throwCube() {
        players[currentPlayer].numberSum += 1+Math.floor(Math.random()*6);
        render();
}
```

anim.js

```
let rot = 0;

async function throwCube() {
        let canvas = id("canvas");
        let ctx = canvas.getContext("2d");
        ctx.imageSmoothingEnabled = false;

        let x=Math.random()*200-100;
        let y=50;
        let z=Math.random()*200-100;
        let r=0;
        let vx=Math.random()*10-5;
        let vy=0;
        let vz=Math.random()*10-5;
        let vr=(Math.random()*10-5)/20;
        let imageNow = Math.floor(Math.random()*6);
        let imageNext = Math.floor(Math.random()*5);
        if(imageNext == imageNow) imageNext = 5;
        let dir=2;
        const halfPi = Math.PI/2;
        while(true) {
                let speed = Math.sqrt(vx*vx+vz*vz)
                rot += speed/50 + 0.5/((1+speed)**3);
                if(rot > halfPi) {
                        rot = rot % halfPi;
                        if(Math.abs(vy) < 2 && Math.abs(y) < 0.1) rot = 0;
                        imageNow = imageNext;
                        imageNext = Math.floor(Math.random()*5);
                        if(imageNext == imageNow) imageNext = 5;
                        dir = Math.floor(Math.random()*4);
                }
                vy -= 1;
                x += vx;
                y += vy;
                z += vz;
                r += vr;
                if(y < 0) {
                        y = 0;
                        vy = vy * -0.8;
                        vx = vx * 0.8 - x / 10;
                        vz = vz * 0.8 - z / 10;
                        vr = Math.random()*speed/100; //vr * 0.8;
                }
                ctx.fillStyle="#888";
                ctx.fillRect(0,0,400,400);
                ctx.save();
                ctx.translate(200+20*x/(60-y), 200+20*z/(60-y));
                ctx.rotate(r);
                ctx.scale(60/(60-y),60/(60-y));
                if(dir == 0) {
                        let s = Math.sin(rot);
                        let c = Math.cos(rot);
                        let x1 = (-1)*c - 1*s;
                        let x2 = ( 1)*c - 1*s;
                        let x3 = ( 1)*c + 1*s;
                        ctx.drawImage(images[imageNow], 20*x1, -20, 20*(x2-x1), 40);
                        ctx.drawImage(images[imageNext], 20*x2, -20, 20*(x3-x2), 40);
                }
                if(dir == 1) {
```

```
                        let s = Math.sin(rot);
                        let c = Math.cos(rot);
                        let x1 = (-1)*c + 1*s;
                        let x2 = ( 1)*c + 1*s;
                        let x0 = (-1)*c - 1*s;
                        ctx.drawImage(images[imageNow], 20*x1, -20, 20*(x2-x1), 40);
                        ctx.drawImage(images[imageNext], 20*x0, -20, 20*(x1-x0), 40);
                }
                if(dir == 2) {
                        let s = Math.sin(rot);
                        let c = Math.cos(rot);
                        let x1 = (-1)*c - 1*s;
                        let x2 = ( 1)*c - 1*s;
                        let x3 = ( 1)*c + 1*s;
                        ctx.drawImage(images[imageNow], -20, 20*x1, 40, 20*(x2-x1));
                        ctx.drawImage(images[imageNext], -20, 20*x2, 40, 20*(x3-x2));
                }
                if(dir == 3) {
                        let s = Math.sin(rot);
                        let c = Math.cos(rot);
                        let x1 = (-1)*c + 1*s;
                        let x2 = ( 1)*c + 1*s;
                        let x0 = (-1)*c - 1*s;
                        ctx.drawImage(images[imageNow], -20, 20*x1, 40, 20*(x2-x1));
                        ctx.drawImage(images[imageNext], -20, 20*x0, 40, 20*(x1-x0));
                }
                ctx.restore();
                if(rot == 0 && Math.abs(vy) < 2 && Math.abs(y) < 0.1) break;
                await waitFrame();
        }
        players[currentPlayer].numberSum += imageNow+1;
        render();
}

let images = [];
for(let i=1; i<7; i++) {
        let img = new Image();
        img.src = "images/"+i+".png";
        images.push(img);
}

async function waitFrame() {
        return new Promise(resolve => requestAnimationFrame(resolve));
}
```

style.css

```
.hidden {
        display: none;
}
.red {
        color: #f22;
        font-weight: bold;
}
table {
        border: 1px solid #444;
        border-collapse: collapse;
        width: 100%;
}
th, td {
        border: 1px solid #444;
        padding: 5px;
}
th {
        background: #888;
}
td {
        background: #aaa;
}
button {
        padding: 5px;
}
body {
        font-family: sans-serif;
}
html, body {
        width: 100%;
        margin: 0;
        text-align:center;
```

```
}
.center {
	display: inline-block;
}
.bold {
	font-weight: bold;
}
```

2. Виконання


Двадцять одно
Оберіть складність
Проста (занадто обережинй)
Середня (занадто обережинй)
Складна
Середня (занадто ризиковий)
Проста (занадто ризиковий)

Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	0	1
Комп'ютер 1	0	1

Купа: 2 Раунд: 1

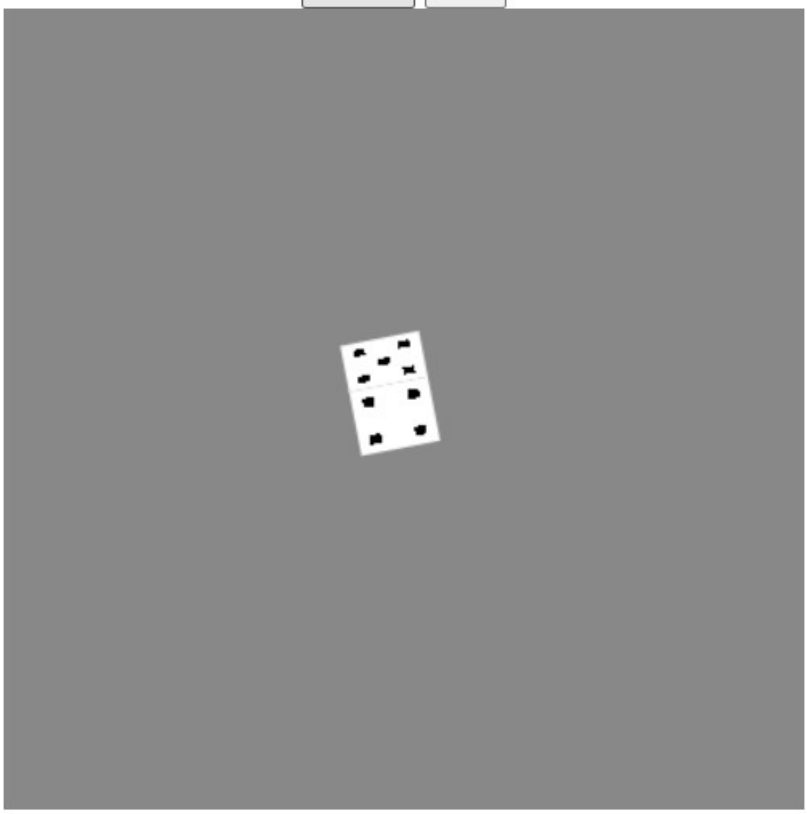
Кинути Далі

Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	4	1
Комп'ютер 1	0	1

Купа: 2 Раунд: 1

Кинути Далі

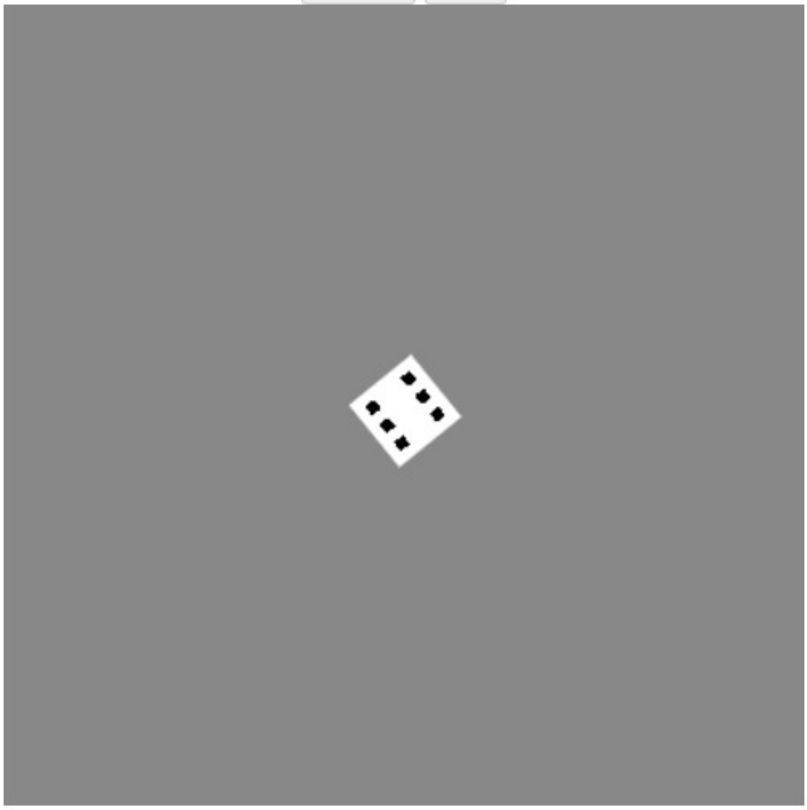


Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	16	1
Комп'ютер 1	0	1

Купа: 2 Раунд: 1

Кинути Далі

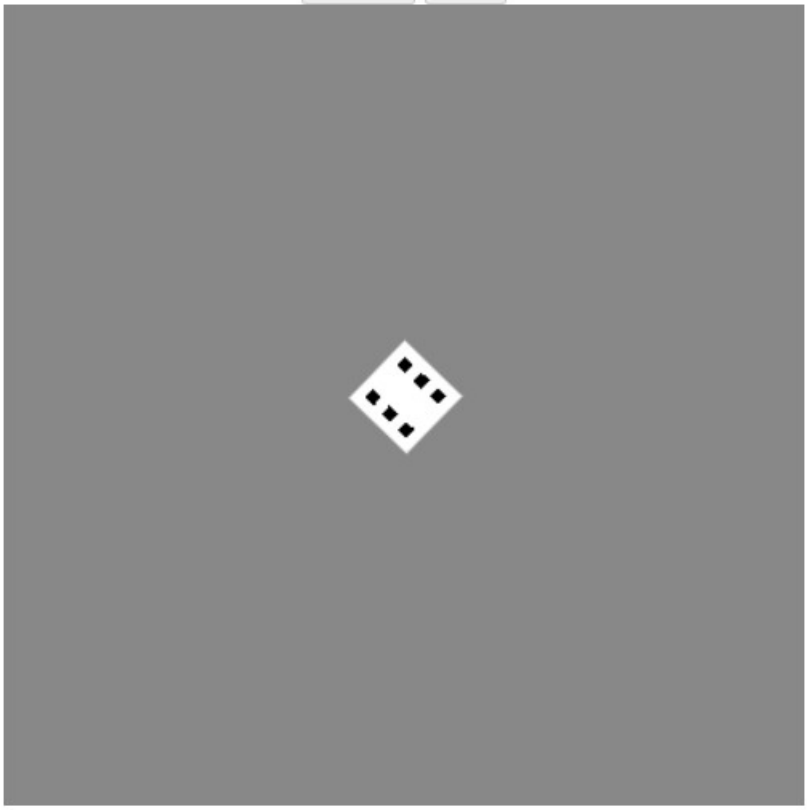


Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	16	3
Комп'ютер 1	22	1

Купа: 0 Раунд: 1

Кинути Далі

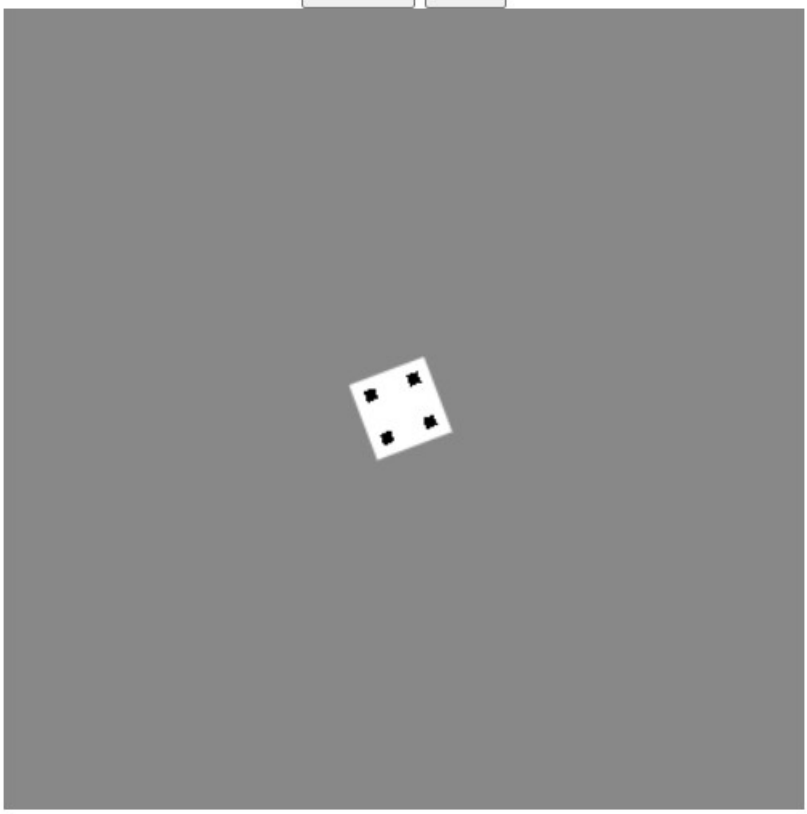


Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	0	2
Комп'ютер 1	17	0

Купа: 2 Раунд: 2

Кинути Далі

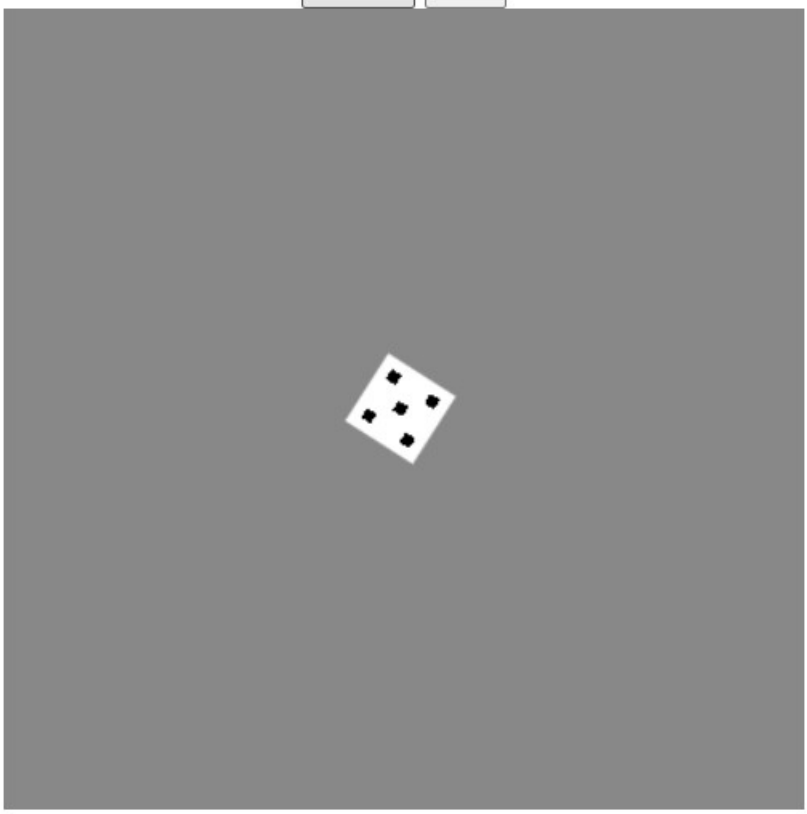


Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	5	2
Комп'ютер 1	17	0

Купа: 2 Раунд: 2

Кинути Далі

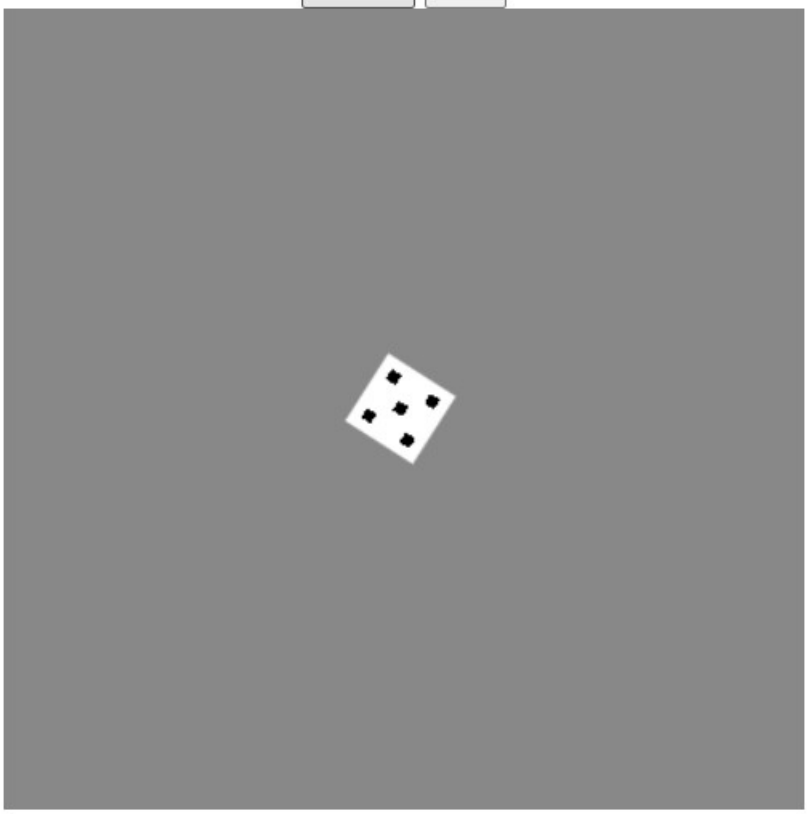


Двадцять одно

Гравець	Сума чисел	Фішок
> Ви	5	2
Комп'ютер 1	17	0

Купа: 2 Раунд: 2

Кинути Далі



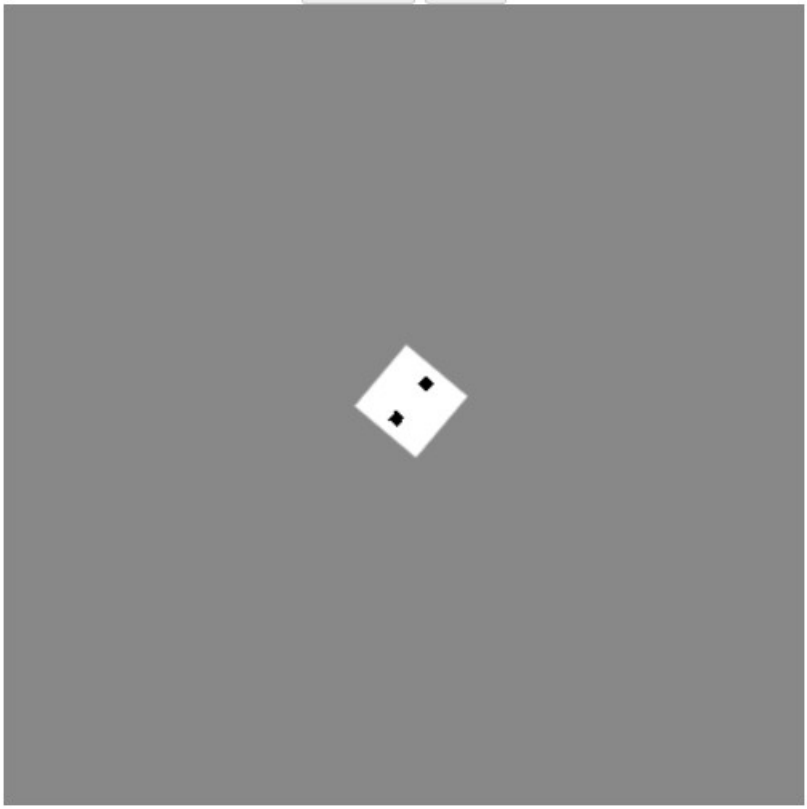
Подтвердите действие

Комп'ютер переміг

OK

Купа: 0 Раунд: 15

Кинути Далі



4. ВИСНОВОК

В рамках даної лабораторної роботи було ознайомлено з алгоритмами мінімаксу та альфа-бета-відсікань. Було реалізовано ігровий додаток для гри "Двадцять одне" на мові програмування javascript. У ньому людина грає проти комп'ютера та дії комп'ютера контролюються на основі алгоритму мінімаксу. А саме, рахуються шанси перемоги, де комп'ютер намагається обирати варіанти з найкращими шансами для себе, при цьому передбачається що людина буде обирати варіанти найгірші для комп'ютера. На основі цього рахуються шанси що використовуються щоб відповісти на питання: чи кидати далі, чи зупинитися.

5. КРИТЕРІЇ ОЦІНЮВАННЯ

При здачі лабораторної роботи до 25.12.2022 включно максимальний бал дорівнює – 5. Після 25.12.2022 максимальний бал дорівнює – 1.

Критерії оцінювання у відсотках від максимального балу:

- програмна реалізація – 95%;
- висновок – 5%.

+1 додатковий бал можна отримати за реалізацію анімації ігрових процесів (жеребкування, роздачі карт, анімацію ходів тощо).