MONTE CARLO SIMULATION PROJECT

CTIS 476

Problem Definition

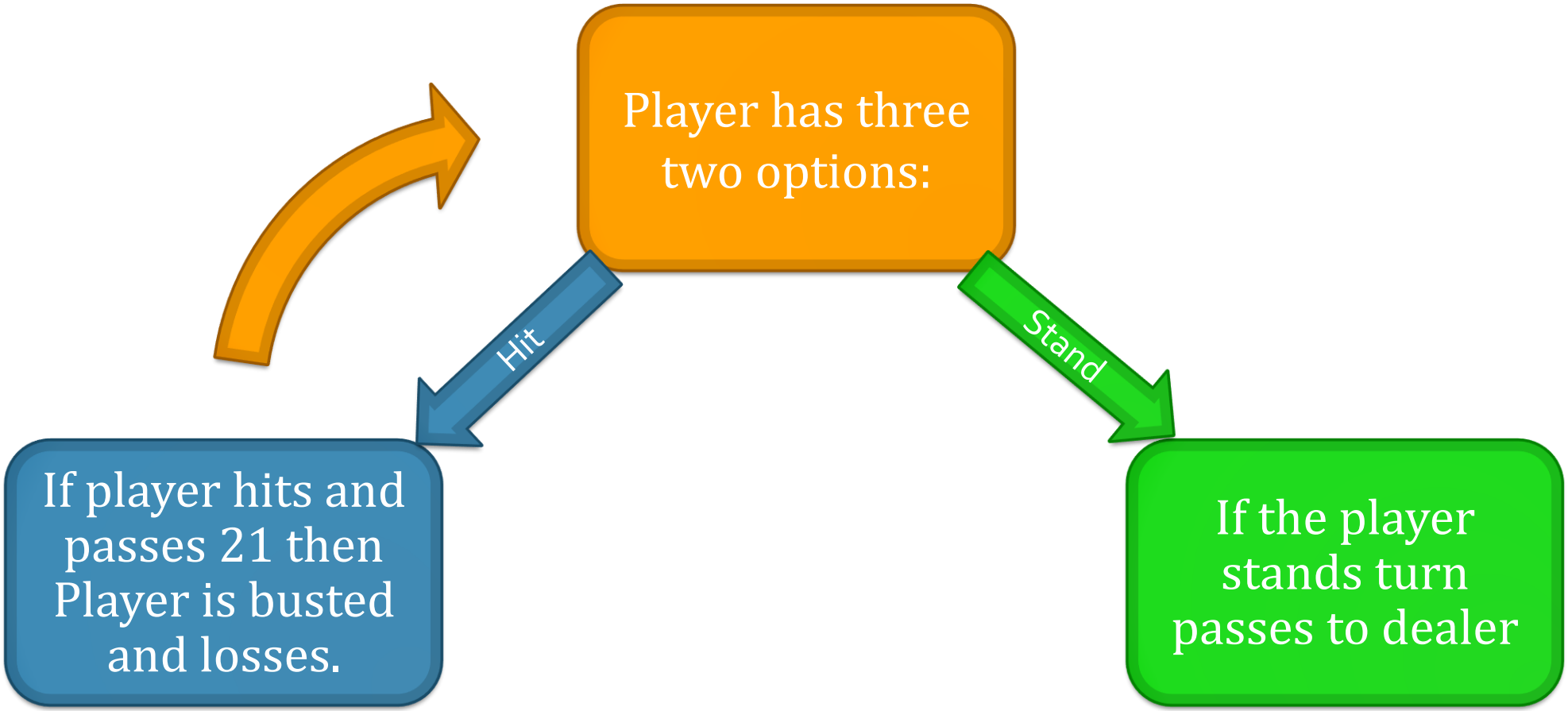
The Rules - Beginning

- Computer is the dealer
- Dealer gets one open card
- Player has two cards

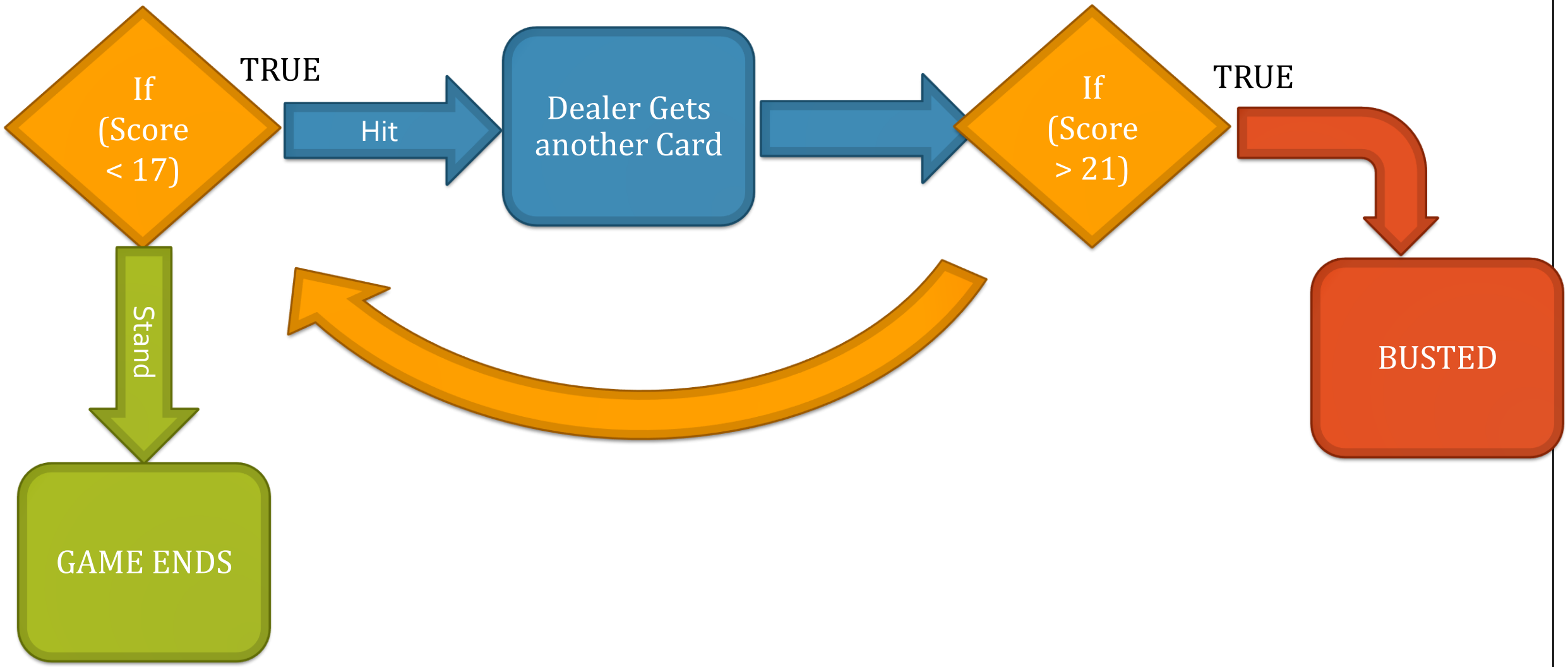
The Rules - Scoring

- Number cards count as their numbers
- Jack, Queen and King is 10 points
- Ace is 1 or 11

The Rules - Player



The Rules - Dealer



The Rules – Winning or Losing

If player is busted, Player loses

If dealer is busted, Player Wins

If the player has more score, Player wins

Else Player loses

The Project

- Build a Monte-Carlo simulation to decide best target score strategy to play the defined game
- You will be provided;
  - SimulationInterface.h
  - Project1.cpp
- You will provide only
  - Simulation.h
  - Simulation.cpp

The Interface

```
enum class CardEnum {
Card_Ace,
Card_2,
Card_3,
Card_4,
Card_5,
Card_6,
Card_7,
Card_8,
Card_9,
Card_10,
Card_Jack,
Card_Queen,
Card_King,
INVALID_CARD
};
```

```
class SimulationInterface {
public:
virtual void initializeSimulation(CardEnum firstCard, CardEnum secondCard, CardEnum dealerCard
virtual void runSim(int numberOfTrialsforEachTarget) = 0;
virtual void getResults(const int &target, int& numberOfWins, int& numberOfLosses) const = 0;
virtual void testHitRandomness(int trials) const = 0;
};
```

The Project1

```
int main()
{
	printf("When asked please enter a card as A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K\n");
	CardEnum dealer = readCard("Please Enter The Dealers Open Card : ");
	CardEnum firstCard = readCard("Please Enter Your First Card : ");
	CardEnum secondCard = readCard("Please Enter Your Second Card : ");
	int trials = readNumOfTrials("Please Enter Number Of Trials : ");

	SimulationInterface* sim = new Simulation();
	sim->testHitRandomness(100000);

	sim->initializeSimulation(firstCard, secondCard, dealer);
	sim->runSim(trials);
	for (int i = 12; i <= 21; i++)
	{
		int wins, losses;
		sim->getResults(i, wins, losses);

		float probability = 0.0f;
		if (wins > 0 || losses > 0)
		{
			probability = ((float)wins) / (wins + losses);
		}
		std::cout << "Target:" << i << " - wins:" << wins << " - losses:" << losses << " - probability of win:"
<< probability << std::endl;
	}

	delete sim;

	system("Pause");
}
```

Sample Run

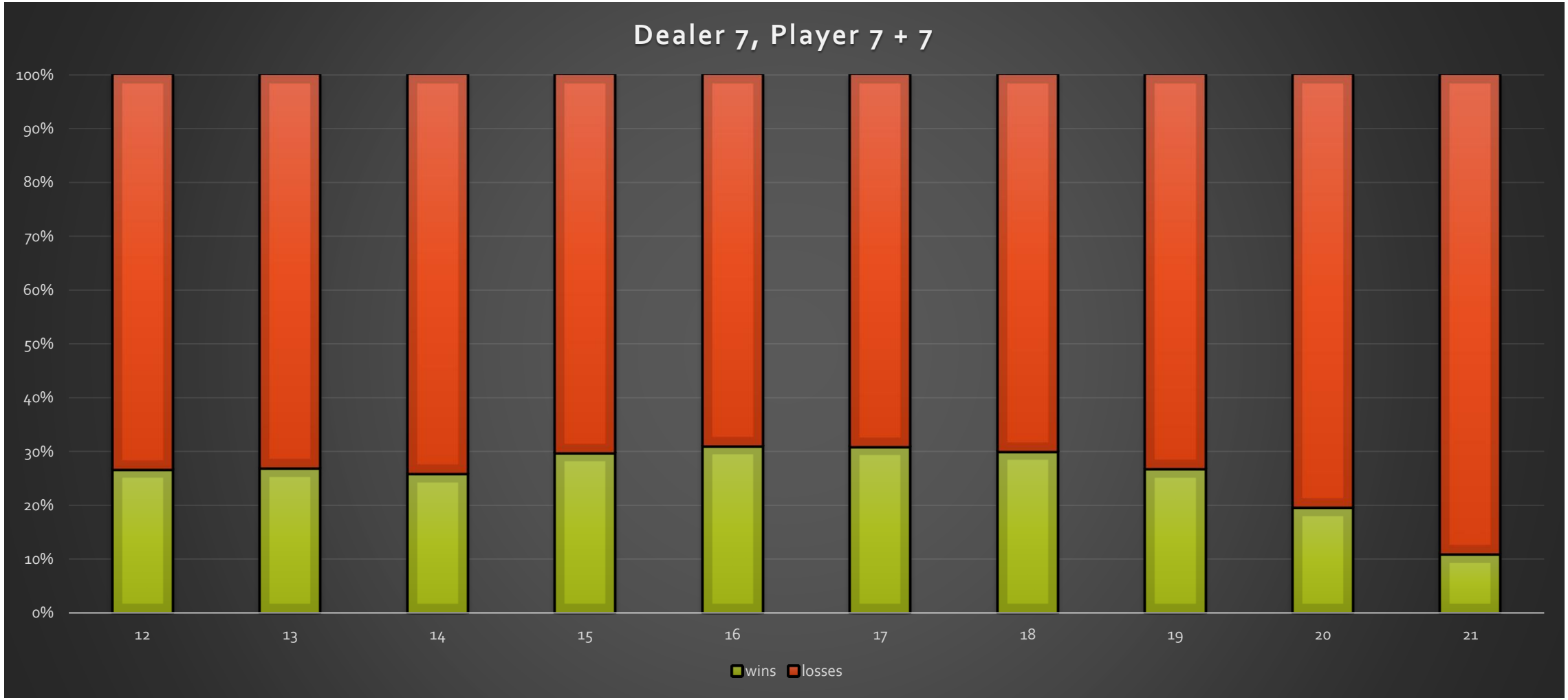
```
When asked please enter a card as A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J,
Q, K
Please Enter The Dealers Open Card : 7
Please Enter Your First Card : 7
Please Enter Your Second Card : 7
Please Enter Number Of Trials : 10000
```

Sample Output

```
Aces  :7698 - %0.07698
2s    :7686 - %0.07686
3s    :7740 - %0.0774
4s    :7741 - %0.07741
5s    :7599 - %0.07599
6s    :7773 - %0.07773
7s    :7659 - %0.07659
8s    :7568 - %0.07568
9s    :7750 - %0.0775
10s   :7784 - %0.07784
Jacks :7667 - %0.07667
Queens:7541 - %0.07541
Kings :7794 - %0.07794
```

```
Target:12 - wins:2669 - losses:7331 - probability of win:0.2669
Target:13 - wins:2689 - losses:7311 - probability of win:0.2689
Target:14 - wins:2590 - losses:7410 - probability of win:0.259
Target:15 - wins:2978 - losses:7022 - probability of win:0.2978
Target:16 - wins:3107 - losses:6893 - probability of win:0.3107
Target:17 - wins:3094 - losses:6906 - probability of win:0.3094
Target:18 - wins:3002 - losses:6998 - probability of win:0.3002
Target:19 - wins:2678 - losses:7322 - probability of win:0.2678
Target:20 - wins:1963 - losses:8037 - probability of win:0.1963
Target:21 - wins:1092 - losses:8908 - probability of win:0.1092
```

Sample Output (Graph)



Evaluation

Criteria	Score
Required files are sent, they are relevant and code compiles without error	30 Points
Project is working for general cases	20 Points
Project is working for extreme case	20 Points
Output is compatible with the expected output Randomness Output Possibility Output	10 Points
Code is clean and readable	10 Points
A chart is provided	10 Points