1. Write a function to print a description of a given card suite (use enum declaration from lab 8).

fn describe(card: Suite) {

match (card) {

Value::Number(number) => println!(“Number: {}”, number),

Value::Face(name) => println!(“Face: {}”, name),

Value::Ace => println!(“Ace”),

Value::Joker => println!(“Joker”)

}

}

1. Write a function to print a description of a given card value (use enum declaration from lab 8).

fn describe(card: Value) {

match (card) {

Value::Number(number) => println!(“Number: {}”, number),

Value::Face(name) => println!(“Face: {}”, name),

Value::Ace => println!(“Ace”),

Value::Joker => println!(“Joker”)

}

}

1. Again using the Suite and Value enums implemented in Lab 8, implement the prototype below for a scoring function for a hypothetical card game with the following rules: The ace of spades is worth 10 points, other aces are worth 8, a joker is worth 7, all face cards are worth five, all other red cards are worth 3, even black numeric cards are worth 2, and odd black numeric cards are worth 1

fn score(suite: Suite, value: Value) -> i32 {

return match (suite, value) {

(Suite::Spades, Value::Ace) => 10,

(\_, Value::Ace) => 7,

(\_, Value::Joker) => 7,

(\_, Value::Face(\_)) => 5,

(Suite::Diamonds | Suite::Hearts, \_) => 3,

(Suite::Spades | Suite::Clubs, Value::Number(num)) if num % 2 == 0 => 2,

(Suite::Spades | Suite::Clubs, Value::Number(num)) if num % 2 == 1 => 1

};

}

1. Complete the simple blackjack AI prototype below. The cards vector is a list of cards currently in the hand. The function should return true to hit and false to stand. For the uninitiated, the dealer will hit on sixteen and stand on seventeen. The rules of blackjack may also be found on Wikipedia. Just consider aces as ones for now.

fn blackjack(cards: Vec<Value>) -> bool {

let mut total = 0;

for card in cards {

total += match card {

Value::Face(\_) => 10,

Value::Ace(\_) => 1,

Value::Number(num) => num,

Value::Joker => panic!(“Jokers aren’t in blackjack”)

};

}

return total < 17;

}