GRQ01 - Programming Rust - Ch 3

Your Name

Chapter 3. Basic Types

1. The first two code samples are equivalent. In the second example, how is the programming language able to tell what `v`'s type is?

Your answer here...

2. There are primitive types that begin with `i` and `u`. What are these types, generally, and what distinction does the `i` and `u` give?

Your answer here...

3. Unlike in Java, any value can have methods. Give an example of a useful method call on a value you could not perform in Java.

Your answer here. If you'd like to represent code, begin each line with a tab as shown here:

```
// This is a code block
let x: u8 = 0;
```

Returning to no indentation brings you back to regular text.

4. To represent the character `n` in Rust, you surround it in single quotes. How do you represent a backslash? Tab? Newline character? Why might this be necessary or useful?

Your answer here...

5. What is a tuple? How do you access the elements of a tuple?

Your answer here...

6. Java uses the `void` keyword to establish the return type of a method that returns nothing. What does Rust's equivalent function return?

Your answer here...

7. Given the code listing below:

```
let i: u8 = 10;
let r: &u8 = &i;
println!("i: {}, r: {}", i, *r);
```

The two ampersands in the declaration and initialization of `r` mean two different things. What is the difference between them? In the `println!` macro call, what is the significance of the asterisk before the `r`?

Your answer here...

8. What is a `Vec`? What kinds of operations can you perform on a `Vec`? What collection type does it closely resemble in Java?

Your answer here...

9. In the Slices section, the variable `sv` is a slice referring to a `Vec`'s elements and `sa` is a slice referring to an array's elements. Both `sa` and `sv` are of the same slice type: `&[f64]`. How is this possible? Hint: Spend some time considering the implications of the memory diagram in Figure 3-2.

Your answer here…

10. What are some of the key differences between types `&str` and `String`? Which type is more appropriate when writing a function to process a string?

Your answer here…