Machine Arithmetic, Characters, and Strings

1-25-2012

Opening Discussion

- Questions about the quiz?
- ACM Meeting
- Minute essay comments:
  - Binary clocks.
  - Is Scala a program within a program?
  - Switching desktops.
  - Programming in the TV/film industry.
  - What is the coolest thing I have ever coded?
  - Google+ e-mails and spam.
  - Jobs understanding complex systems.

Manual Binary Conversion

- Going from binary to decimal is done by simply adding the values of the positions for the 1s.
- We'll describe two methods of going from decimal to binary.
  - Repeated subtraction of “largest power of 2”.
  - Repeated division by 2. This one is probably easiest.

Binary Addition

- Adding binary numbers is very easy. Just do the long addition that you are used to.
- You will carry a lot more frequently because anything above 1 causes a carry.
- Let's run through some examples.
- Consider implications of fixed precision.

Negative Numbers

- We don't have a – in the computer for negative numbers. All we have are 1 and 0. So how do we make negative numbers?
- Remember the definition of negative numbers as additive inverse.
  - a+(-a)=0
- We want to preserve this to keep addition simple.
- This gives us 2s-compliment numbers.

Binary Multiplication

- Multiplying binary numbers works just like long multiplication with decimals, but easier.
- My only recommendation is you only add two numbers at a time and take it in steps.

Hexadecimal

- Binary is unwieldy for humans because of the large number of digits.
- Hexadecimal (base 16) is commonly used because it converts nicely to binary, but has few digits.
- Four bits is a hex digit. Start at the right and group bits by 4.
- Use letters A-F for numbers 10-15.
- Hex literals start with 0x
- toHexString

Octal

- Octal (base 8) is less common than hex, but not uncommon.
- Group bits into groups of three.
- Octal literals and toOctalString().

The math Object

- For other math functions use methods on the math object.
- For example, use math.sqrt() to take the square root of a number.

Characters

- The Char type represents a single character in Scala.
- The literal for Char has the letter that you want in single quotes.
- The Char is stored in the computer as a 16-bit unsigned integer encoded in Unicode.
- Unicode has the alphabet of every written language in it.
- You can convert to an Int to see the numeric values of characters.

Escape Characters

- Not all characters can be easily entered. For things you can't nicely type, use escape characters.
  - \n – for a new line
  - \t – for a tab
  - \” - to get a double quote
  - \' - to get a single quote
  - \\ - to get a backslash

Strings

- We have seen the String type and that re represent String literals by putting characters in double quotes.
- Escape characters can also go inside of normal strings.
- Strings have many methods. We can see the basics using tab completion. (If we put in some extra parentheses.)

Raw Strings

- There are some situations when using escape characters is a pain.
- For this, use triple double quotes to make a raw string.
- Anything you type between the triple double quotes will go into the string.
- They can span multiple lines even.

Variables

- It is very common to want to represent values with names.
- A variable is a name that we use to represent a value.
- In Scala we can declare variables using val or var.
  - val *name*:*Type* = *expression*
  - var *name:Type* = *expression*
- A val can't change it's value, a var can.
- The colon and type are generally optional.

Tuples

- Another type in Scala is the Tuple type.
- A tuple has comma separated values in parentheses.
- They give us a way to handle a fixed set of associated values.
- Assignment into a tuple does pattern matching.

Minute Essay

- What question do you have about the topics we have looked at?