CS16, 10W, Handout to go with H04 (Operators, Operands) (printable PDF)

Available online at: http://www.cs.ucsb.edu/~pconrad/cs16/10W/homework/H04/handout

The assignment is available at http://www.cs.ucsb.edu/~pconrad/cs16/10W/homework/H04

Please obtain the official textbook for this course by Delores Etter. **You will start having homework assignments in that book next week.** (I've delayed this one additional class period since the book is not yet on reserve at the library.)

For today, this handout is your reading assignment to go with H04. If you have the official textbook, sections 2.1 and 2.1 will help you complete this assignment.

Homework H04 is about the following terminology:

- operator
  - unary operator
  - binary operator
- operand
  - left operand
  - right operand

It is also a check on whether you understand the concept of operator prededence, and the notion of type.

Here are several sample questions and answers, and an explanation of each. From those, and the material in lecture, you should be able to answer the homework problems.

Question: Given the expression 2 + 3 * 5

1. What is the value of the expression?
2. What is the type of the expression?
3. How many binary operators are in this expression?
4. What is the left operand of the * operator?
5. What is the right operand of the + operator?

Answers:

1. The value is 17. This is because the first operation performed is the 3 * 5, which results in an int value 15. The next operation performed is 2 + 15, resulting in 17, which is an int.
2. The expression is of type int (see above)
3. There are two binary operators, + and *. A binary operator is an operator that takes two operands. Operands are the things the operator operates on to produce a result.
4. The left operand of the * operator is 3. (The right operator would be 5)
5. The right operand is actually *the result of (3 * 5)*. I would also accept the answer "15".
   Note that 3 is not a correct answer, because we do not add 2 and 3.
   (The left operand of the + is of course 2).

Please turn over for more

Continued from other side

Here's another example:

Question: Given the expression 5 == (2 + 2)

1. What is the value of the expression?
2. What is the type of the expression?
3. How many binary operators are in this expression?
4. What are the operands of the == operator?

Answer:

1. The value is 0. The first operation performed is the 2 + 2 which results in an int value 4. The next operation performed is 5 == 4, which results in the int value 0 (boolean values are treated as int values in C, with 0 representing false, and 1 representing true.)
2. The expression is technically of type int, although I might also accept "boolean".
3. There are two binary operators, == and +. A binary operator is an operator that takes two operands. Operands are the things the operator operates on to produce a result.
4. The operands of the == operator are 5 (which is the left operand) and "the result of (2+2)" (which is the right operand).

Here's another example:

Question: Suppose that x is a variable of type int, with the value 6, and y is variable of type int with the value 4.

In the expression x = -y

1. What is the value of the expression?
2. What is the type of the expression?
3. How many binary operators are in this expression?
4. How many unary operators are in this expression?
5. What is the operand of the unary operator?
6. What are the operands of the binary operator

Note: that I have left out the ; deliberately here—this is an expression, not necessarily a statement. This expression could be turned into an assignment statement, by adding a semicolon, like this:

```
 x = -y;
```

But this expression could also appears in other contexts such as these. These are both a little strange, but they are legal C syntax. They help to illustrate the idea of an expression vs. the idea of a statement.

```
if (x = -y)

...

printf("%d\n", x = -y);
```

Answer:

1. The value is -4. Here's how that happens. We have an assignment, so we first evaluate the right hand side. The right hand side contains the unary operator minus, i.e. - (that is, an operator with only one operand). That is applied to the operand y, which results in the value -4 (we fetch the value of y, namely 4, out of memory and then apply the - sign to negate the value.).

   Then, the expression x = -4 is evaluated. The assignment operator = is treated as a binary operator with left operand x, and right operand -4 (the result of applying the unary operator - to the value of y). The = operator results in the left operand being assigned the value of the right operand, and then returns the value assigned as the result of the expression. So the value is -4
2. The type is int
3. There is one binary operator, namely =
4. There is one unary operator, namely -
5. The unary operator is -, and its operand is y
6. The binary operator is =, and its operands are x (left operand), and the result of -y (right operand)

With those examples, you should be able to complete homework H04.

End of H04 handout