

Example Solutions for Programming Languages G22.2110 Summer 2007 hw08

Assigned Th 7/12/2007, due We 7/18/2007 at 1pm.

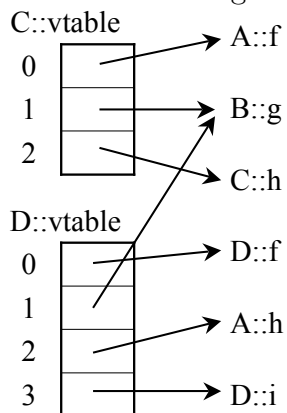
These are example solutions. Please keep in mind that often, there is not just one correct solution to a question. If you come up with different answers to the homework, then it may be that both your answers and these answers here are correct. Of course, these answers here may also contain mistakes; if you spot some, please let us know so we can correct them.

Reading Assignments

- For lecture on 7/12/2007: Scott 8.1, 8.2.0, 8.3.0-8.3.1, 8.5.0-8.5.2
 - For lecture on 7/19/2007: Scott 7.2.4 (on CD);
Cumming (<http://www.dcs.napier.ac.uk/course-notes/sml/manual.html>)
-

Homework Assignments

1. Virtual method dispatch (10 points)
Consider the following virtual method tables:



Write Java classes A, B, C, and D such that C and D have the vtables shown above. Make sure that your code causes no compiler errors.

Example solutions

```
class A {
  void f() { System.out.println("A::f"); }
  void g() { System.out.println("A::g"); }
  void h() { System.out.println("A::h"); }
}
class B extends A {
  void g() { System.out.println("B::g"); }
}
class C extends B {
  void h() { System.out.println("C::h"); }
}
class D extends B {
  void f() { System.out.println("D::f"); }
  void i() { System.out.println("D::i"); }
}
```

2. SML (0 points)

Start teaching yourself SML by doing the following:

- 2a. If possible, find peers (other students who want to learn SML together with you) and gurus (people who already know SML, whom you can ask questions when you get stuck).
- 2b. Make sure you have access to the SML/NJ interpreter. Ask your guru if you are having problems with this step.
- 2c. Read the SML tutorial (see reading assignment above). Along the way, try things out with the SML interpreter that you installed in Step b.
- 2d. Familiarize yourself with the structure of the online SML documentation, so you can find information quickly when you need it. In particular, find the URLs for the Standard ML Basis Library and for the Harper tutorial.

3. Parameter passing modes (20 = 5 + 5 + 5 + 5 points)

Consider the following Ada program:

```
with Ada.Text_IO;
procedure Main is
  I : Integer := 1;
  A : array(1..2) of Integer := (1, 1);
  procedure P is begin
    Ada.Text_IO.Put_Line(
      Integer'Image(I) & Integer'Image(A(1)) & Integer'Image(A(2)));
```

```

end P;
procedure Q(J: in out Integer; K : in out Integer) is begin
  I := 2;
  K := 3;
  Ada.Text_IO.Put_Line(Integer'Image(J) & Integer'Image(K));
  P;
end Q;
begin
  Q(I, A(I));
  P;
end Main;

```

3a. (5 points) What does the program print if “in out” means call by value?

Example solutions

```

1 3
2 1 1
2 1 1

```

Explanation:

- at the time of the call, the actuals get evaluated and their r-values (1 and 1) get copied to the formals J and K
- the assignments to I and K affect only I and K, but not J or A
- at the time of the return, nothing gets copied back

3b. (5 points) What does the program print if “in out” means call by reference?

Example solutions

```

2 3
2 3 1
2 3 1

```

Explanation:

- at the time of the call, the actuals get aliased with the formals, so J becomes an alias for I and K becomes an alias for A(1)
- the assignment to I also affects its alias J, so both become 2
- the assignment to K also affects its alias A(I), so both become 3
- at the time of the return, nothing gets copied back

3c. (5 points) What does the program print if “in out” means call by value-result?

Example solutions

```

1 3
2 1 1
1 3 1

```

Explanation:

- at the time of the call, the actuals get evaluated and their

```
r-values (1 and 1) get assigned to the formals J and K
- the assignments to I and K affect only I and K, but not J or A
- at the time of the return, the formals get copied back into the
  actuals: I becomes J, which is 1, and A(1) becomes K, which is 3
```

3d. (5 points) What does the program print if “in out” means call by name?

Example solutions

```
2 3
2 1 3
2 1 3
```

Explanation:

- at the time of the call, the formals get bound to unevaluated actuals
- the assignment to I proceeds as usual, changing I to 2
- the assignment to K is really an assignment to A(I), which is A(2), because I has changed in the meantime
- at the time of the return, nothing gets copied back

4. SML (10 = 2 + 3 + 2 + 3 points)

Continue teaching yourself SML by reading the example code from

<http://www.cs.nyu.edu/courses/summer07/G22.2110-001/hw08-sml-example.txt>

4a. (2 points) Run the program. Answer the prompts as follows:

```
Please enter a floating point number, or an empty line to end the list: 2.4
Please enter a floating point number, or an empty line to end the list: -2.1
Please enter a floating point number, or an empty line to end the list: 5.0
Please enter a floating point number, or an empty line to end the list: 4.23
Please enter a floating point number, or an empty line to end the list:
```

What does the program print?

Example solutions

```
arithmetic mean: 2.3825, standard deviation: 2.75488997058
```

4b. (3 points) What is the type of function `real_square`?

Example solutions

```
real -> real
```

Explanation: `real_square` is a function that takes one argument of type `real`, and returns a result of type `real`.

4c. (2 points) Remove the “: `real`” from the definition of function `real_square`. What error message does the compiler report?

Example solutions

operator and operand don't agree [tycon mismatch]

operator domain: int * int

operand: int * real

in expression: real_square (h - m) + standard_deviation_helper (t,m)

4d. (3 points) Briefly explain the error message from Step c.

Example solutions

A "tycon mismatch" occurs when the type inference engine encounters conflicting constraints. In this case:

- the operator is "+"

- the left operand is "real_square (h - m)"

- the right operand is "standard_deviation_helper (t,m)".

Due to the change to real_square, the type of the left operand is int, whereas the type of the right operand is real. But SML does not allow mixed operands for +; they must be either both int or both real. In this case, the error message indicates that the SML interpreter was expecting that both operands are int (operator domain: int * int).

5. Parameter passing modes (10 = 3 + 7 points)

Consider the following C++ program:

```
#include <stdio.h>
void integer_divide(int x, int y, int& quotient, int& remainder) {
    quotient = x / y;
    remainder = x % y;
}
int main(int argc, char** argv) {
    int a=7, b=3, q, r;
    integer_divide(a, b, q, r);
    printf("%d / %d = %d + %d / %d\n", a, b, q, r, b);
    return 0;
}
```

5a. (3 points) What does this program print?

Example solutions

7 / 3 = 2 + 1 / 3

5b. (7 points) Translate the function integer_divide from C++ to C. The C version should have the signature

```
void integer_divide(int x, int y, int* quotient, int* remainder)
```

It should be line-by-line equivalent to the original C++ version.

Example solutions

```
#include <stdio.h>
void integer_divide(int x, int y, int* quotient, int* remainder) {
    *quotient = x / y;
    *remainder = x % y;
}
int main(int argc, char** argv) {
    int a=7, b=3, q, r;
    integer_divide(a, b, &q, &r);
    printf("%d / %d = %d + %d / %d\n", a, b, q, r, b);
    return 0;
}
```

<http://www.cs.nyu.edu/courses/summer07/G22.2110-001/hw08-example-solutions.pdf>

Total points: 50.