Question 5 (15%). A function name is ***overloaded*** if different argument types invoke distinct functions. For example, in most languages, + is overloaded because 2+2 invokes integer addition but `2.0+2.0` invokes floating-point addition.

What is the difference between overloading in C++ and overloading in Haskell?

Question 6 (20%). The string p = `"cde"` is a ***substring*** of the string s = `"abcdefg"` because all of the characters of p occur in s in the same sequence, and with no gaps, as they do in p. By convention, the empty string `""` is a substring of every string and has no substrings except the empty string. A programmer proposes the following definition for a function `substring`:

```
substring "" _ = True
substring _ "" = False
substring s@(x:xs) (y:ys) =
    if x == y
        then substring xs ys
        else substring s ys
```

(a) [5%] Suppose that `""` is changed to `[]` in the base cases of the definition. What effect would this have on the function?

(b) [5%] Show by examples that this definition is incorrect. (The base cases are correct: the error is in the general case.)

(c) [10%] Give a correct definition.

General Notes

Base case and general case. The following terminology is used in several questions. In the function definition

```
fib 0 = 1
fib 1 = 1
fib n = fib(n - 1) + fib(n - 2)
```

the first two lines (`fib 0` and `fib 1`) are called ***base cases*** and the third lines (`fib n`) is called the ***general case***.

Take and drop. If `x` is a list, then `take n x` returns the first `n` elements of `x` and discards the rest, and `drop n x` discards the first `n` elements and returns the rest.

COMP 348 Principles of Programming Languages

Fall 2002

Midterm 1 8 October

Question 1 (20%). The function `f` is defined in a Haskell program as follows:

```
f :: Ord a => a -> [a] -> [a]
f x [] = [x]
f x p@(y:ys)
  | x <= y    = x : p
  | otherwise = y : f x ys
```

(a) Explain the meaning of the first line (the type declaration).

(b) Explain the meaning of `p@(y:ys)`.

(c) Explain the meaning of the lines beginning with "|".

(d) Describe, by means of examples or otherwise, what the function does.

Question 2 (15%). The function `snip` is defined by

```
snip m n = drop m . take n
```

Describe the effect of applying `snip m n` to a list.

Question 3 (15%). The general cases of the definitions for the standard functions `take` and `drop` are:

```
take n (x:xs) = x : take (n-1) xs
drop n (x:xs) = drop (n-1) xs
```

Give suitable base cases for `take` and `drop`, using examples to show how they work.

Question 4 (15%). Here is a short Haskell program:

```
s = take 5 (from 7)
  where
    from n = n : from (n+1)
```

(a) What is the value of "`from 7`"?

(b) What is the value of "`s`"?

(c) Explain why the program doesn't loop forever.

Ideas for Phase Two Presentation

As we mentioned in our last presentation:

What we plan on implementing

How we decided to divide the teams: the rationale for this choice

Developed use cases: describing functionality of system: Why this was important

EXPLANATION:

Diagram of how Parser Render WML Editor are related:

- Design of Parser-Render-can show diagrams
  What has been done so far
  Any difficulties encountered:

- Talk about issues we have faced during design and implementation

- Explanation of what has been done in WML Editor

WHERE WE GO FROM HERE:

- Have to show new task allocation and deadlines…