

Homework 6

Due Sunday, April 10 by 10:00pm

Handout 16
CSCI 334: Spring 2022

Turn-In Instructions

Turn in your work using the Github repository assigned to you. The name of the Github repository will have the form `cs334lab6_<your user name>`. For example, my repository would be `cs334lab6_dbarowy`. You should have received an invite to commit in the repository in your email. If you did not receive this email, please contact me right away!

For this assignment, your completed submission should consist of three parts.

1. A \LaTeX source file called `lab6.tex` and pre-built PDF called `lab6.pdf` in a folder called `pset` that contains solutions to any written questions. A \LaTeX template is provided to help you get started. Be sure to `git add` all necessary files (e.g., images) if your \LaTeX depends on it. For full credit, please ensure that your \LaTeX file builds without error.
2. For each coding question in this assignment, create a project directory. For example, the source directory for question 1 should be in a folder called “q2”. You should be able to `cd` into this directory and then run the program by typing the command “`dotnet run`”. Each program should be split into two pieces: a “`Program.fs`” file that contains the `main` method and associated program-startup routines (like argument parsers), and another “`Library.fs`” file that contains the function(s) of interest in the question. All library code should be in a module named “`CS334`”. Be sure to provide usage output (defined in `main`) for all programs that require arguments. For full credit, your program should both build and run correctly.
3. A \LaTeX source file called `project.tex` and pre-built PDF called `project.pdf` in a folder called `project` that contains project-related writing. A \LaTeX template is provided to help you get started.

Honor Code

This is a partner lab. You may work with another classmate if you wish, and you may co-develop solutions. Remember: although you can work on code together, you must each independently write up and submit your solution. No code copying is allowed. **Be sure to tell me who your partner is** by committing a `collaborators.txt` file to your repository (2 points).

This assignment is due on Sunday, April 10 by 10:00pm.

Sanity Check: Students sometimes submit incomplete assignments, accidentally forgetting to run `git add` for all of their files. Fortunately, there is an easy way to make sure that this does not happen to you. Before you are done, `git clone` your repository to a new folder and then try building/running everything. It only takes a couple minutes and can spare you from headaches later on.

Reading

1. (Required) “A Brief Introduction to F#”
2. (Required) “A Slightly Longer Introduction to F#”

Problems

Q1. (10 points) F# Types

Explain the F# type for each of the following function declarations:

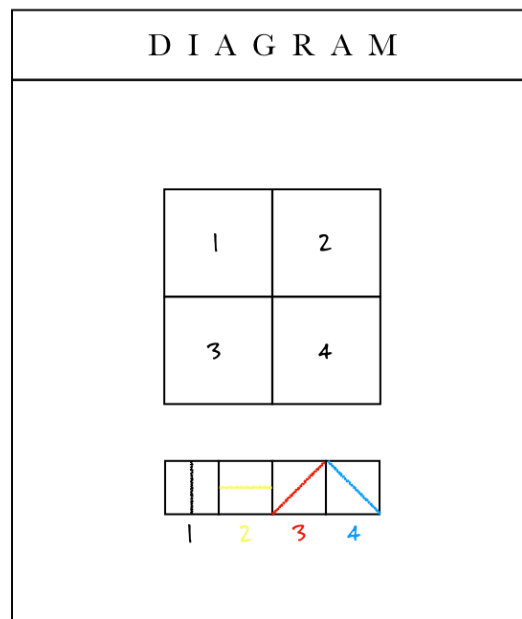
- (a) `let a x y = x + y / 2.0`
- (b) `let b(x,y) = x + y / 2.0`
- (c) `let c f = fun y -> f y`
- (d) `let d f x = f (f x)`
- (e) `let e x y b = if b y then x else y`

Since you can (and should) simply type these expressions into the `fsharp` interpreter to determine the type, be sure to write a brief explanation to demonstrate that you understand why the function has the type you give. A good explanation says something like “a is a function that takes as input ... and it returns as output ...”, being careful to explain what each of those inputs and outputs are. In particular, be sure to explain why each input and output has the type inferred by `fsharp`.

Be sure to answer this question in your `LATEX` file.

Q2. (10 points) Is this a program?

Please “evaluate” the following “program.” Take a photo of the result with your phone and add it to your `LATEX` submission.



A square divided horizontally and vertically into four equal parts, each with a different color and line direction.

Red, yellow, blue, black pencil

Q3. (8 points) Sum of Squares

Define a function `sumSquares(n: int) : int` that, given a nonnegative integer n , returns the sum of the squares of the numbers from 1 to n :

```
> sumSquares 4;;  
val it : int = 30
```

```
> sumSquares 5;;  
val it : int = 55
```

You may define this function recursively (`let rec sumSquares ...`) or by using a `fold` (if you're looking for a challenge).

The project directory for this question should be called “q3”. You should be able to run your program on the command line by typing, for example, “`dotnet run 4`”.

Q4. (10 points) List duplication

Define a function `listDup(e: 'a)(n: int) : 'a list` that takes an element, e , of any type, and a non-negative number, n , and returns a list with n copies of e :

```
> listDup "moo" 4;;  
val it : string list = ["moo"; "moo"; "moo"; "moo"]
```

```
> listDup 1 2;;  
val it : int list = [1; 1]
```

```
> listDup (listDup "cow" 2) 2;;  
val it : string list list = [["cow"; "cow"]; ["cow"; "cow"]]
```

The project directory for this question should be called “q4”. You should be able to run your program on the command line by typing, for example, “`dotnet run moo 4`”.

Q5. (20 points) Peanut Butter and Jelly

`pbj(n: int) : string` is a function that returns a sentence, which is a **string** composed of a sequence of words. n is a positive, nonzero integer supplied by the user. *For each consecutive integer* between 1 and n inclusive, `pbj` either appends the empty string to its output or a word. A word is a string that contains the substring “**peanutbutter**” if n is evenly divisible by 3 or “**jelly**” if n is evenly divisible by 5. Each outputted word in the final sentence should be separated by a single space character. The entire sentence must end with the word “**time**” and a “.” character. Finally, use recursion to solve this problem.

Here are the first ten outputs of the program.

```
$ dotnet run 1
time.
$ dotnet run 2
time.
$ dotnet run 3
peanutbutter time.
$ dotnet run 4
peanutbutter time.
$ dotnet run 5
peanutbutter jelly time.
$ dotnet run 6
peanutbutter jelly peanutbutter time.
$ dotnet run 7
peanutbutter jelly peanutbutter time.
$ dotnet run 8
peanutbutter jelly peanutbutter time.
$ dotnet run 9
peanutbutter jelly peanutbutter peanutbutter time.
$ dotnet run 10
peanutbutter jelly peanutbutter peanutbutter jelly time.
```

The project directory for this question should be called “q5”. You should be able to run your program on the command line by typing, for example, “`dotnet run 10`”.

Q6. (40 points) **Project Brainstorming**

For your final project, you will design and implement a programming language. For this part of the assignment, you should think of three different possible final projects you might explore. Feel free to let your imagination run wild.

If you find the notion of creating your own programming language daunting, don't worry. In future project checkpoints, you will be given the option of choosing a "default" project whose scope will be a little less open-ended. For now, throw caution to the wind with the assurance that at this stage any idea you dream up is non-binding.

Note that a programming language need not be a so-called general purpose programming language. A general purpose programming language is equivalent in expressive power to the lambda calculus or a Turing machine, meaning that it can be used to compute anything. Instead, I encourage you to explore domain specific programming languages, or DSLs. DSLs are usually tailored toward solving a specific problem.

You have probably used a DSL without even realizing it. Some example DSLs are:

- (a) `make`
- (b) GraphViz
- (c) Hypertext Markup Language (HTML)
- (d) Extensible Markup Language (XML)
- (e) Structured Query Language (SQL)
- (f) Markdown
- (g) \LaTeX
- (h) Scalable Vector Graphics (SVG)
- (i) Csound

If you have a personal itch, scratch it. For example, I often have to grade student work. Therefore, the grading rubrics I supply to my teaching assistants are actually written in a domain specific language I designed called `tabulator`.

Be creative! I love music and art, especially abstract art. Can you make a language that generates music? Could you make a language that creates art? Some of your former classmates have designed languages that have done precisely these things and more.

For each potential language, describe

- (a) What problem it solves. What does it do? When I have trouble focusing on this part, it often helps me to think in terms of *inputs* and *outputs*.
 - i. *Programs*, written in the language you design, are the inputs.
 - ii. The output is what happens *when one of those programs is evaluated*.
- (b) What a sample program in the language might look like. Feel free not to be constrained by reality at this point. Just imagine that your programming language already exists. Write two sample programs for your language.

Be sure to submit your project brainstorm in a file called "`project.tex`" along with a pre-built "`project.pdf`" in a folder called "`project`".

Example

If you still feel like you're struggling to understand what this assignment asks of you, here is an example using a language that I actually designed.

FLARE is a language that helps users extract data from spreadsheets. FLARE extends regular expressions—which match text patterns—to two dimensions, specifically spreadsheets. Where a regular expression might return all of the substring matches in a string, a FLARE program returns all of the cells that match in a spreadsheet. Importantly, a user can provide a sequence of cell patterns that extract “rows.”

Why is that useful? Often people use spreadsheets as a kind of database. Later, when they want to use those spreadsheets as input to other programs, they find that the layout they chose prevents them from computing things easily. Many tools expect data to be neatly arranged in columns, and when it's not, users are stuck. Using FLARE, a user can “rearrange” their data to work with those tools.

For example, take the following spreadsheet. Suppose we want to extract the four-tuple (country,value,year,note). One possible tuple is highlighted in red. Another is highlighted in blue.

	A	B	C	D	E	...	R
1		value	year	value	year		Comments
2	Albania	1,000	1950	930	1981		FRA 1
3	Austria	3,139	1951	3,177	1955		FRA 3
4	Belgium	541	1947	601	1950		
5	Bulgaria	2,964	1947	3,259	1958		FRA 1
6	Czech ...	2,416	1950	2,503	1960		NC

The FLARE program,

```
<Harvest,-w>: u+/value/  
[r<Date,-w>, l*<Country, w>]
```

when run on the above spreadsheet, produces the following output. It extracts all of the matching tuples, including the ones I highlighted before in red and blue.

	A	B	C	D
1	Albania	1,000	1950	FRA 1
2	Albania	930	1981	FRA 1
...				
5	Austria	3,139	1951	FRA 3
6	Austria	3,177	1955	FRA 3
...				
9	Belgium	541	1947	
10	Belgium	601	1950	

Note that I describe FLARE just so that you can see what a project idea might look like. FLARE was a part of a multi-year project and it has a level of sophistication that I would not ask any student to reproduce. So do not worry at this phase about making your project look polished or feature-rich. Just try to think of something useful or fun. Find something that speaks to *you*. When I am wondering what to work on next, I usually play a little game with myself that goes something like: “I really hate it when computers make me do _____.” After a minute or two, I can usually turn that complaint into an idea for a new programming language. Despite appearances, I really loathe computers, so I basically never run out of ideas.