CSCI 308, Spring 2021
Programming Languages
Professors Wittie and Kelly
Research Paper

Read this whole document first before you begin the assignment.

Your assignment is to look at a new programming language and **write a report** explaining its design choices. You will also **produce software** written in your language and a short video to demo your software.

The Report

Preliminary portions of your report are due in phases as described below. In the phases, you will choose a language, write a hello world, discuss the paradigms for your language, and plan some software. You will copy (edited as needed) the hello world and paradigms phases into your final report.

Your report will demonstrate a **hello world program** and **discuss the paradigms of your language** and also answer **23 design questions** for a total of 25 parts.

The due date of these phases and the overall report is on the schedule.

Each design question is graded as

- full credit for a good answer, a good argument/code, and a citation
- partial credit for a problematic argument/code or a bad answer or a missing needed citation or its unclear what topic/question you are answering
- no credit for multiple things wrong with the question, or a completely useless or missing argument/code
- Your report needs to have a consistent, readable layout. It needs to have correct grammar and spelling. Your goal is to be informative and concise rather than long winded.
- Type your report and submit it in PDF format.
- Do not use a black background, it makes your report unprintable.
- Format your questions and answers so they are easy to find and refer to, and are numbered.
- Embed your code directly in your report.
- Your citations may be done as embedded citations or footnotes or a bibliography as long as you are consistant.
- Put your name and your language's name at the top/start of your report.
- Include your (copy them in) hello world and paradigms questions (edited if needed) in your report.

Your report must be submitted as a **pdf**. Your report has a **15** page limit including all code examples.

The report is worth 75% of the overall project.

The Software

Your software must be written **by you** in your language and must be an example of something that is appropriate to your programming language. You would not write a website in LISP. You would not write software with a GUI in C. You would not answer logic questions about your family tree in Java. You would not do intensive math computations in Bash. **Your project may not be a calculator!**

Your software will meet the following requirements

- Do something meaningful in your language that exemplifies the usage of your language (something your language does well).
- At least 300 lines of code (comment-only lines don't count).
  Strictly functional languages must be at least 150 lines of code due to their brevity.
- At most 1500 lines total including comments and blank lines.
- Use minimal, but existing, comments.
- Use good coding style for your language.
- Seriously consider putting a copyright statement at the top of your code file(s).

Using other people's code. Its fine if your project makes use of libraries or needs a portion of support code not written by you as long as you cite the source for those parts. **Those parts do not count towards your code length and you only get credit for software features your own code adds.**

Re-using your own code. You may be taking other classes which motivate your software. Bucknell's policy on re-using work from class to class is that it is only permitted if both Profs are explicitly ok with it. I am explicitly ok with it as long as it is a solo project done by you. You can always come ask me for permission if its part of a group project elsewhere. If you are doing this with another class you are in, make sure to get the other Profs "ok" too. I am also ok with it if any part of this ends up re-used in a future class's project.

Ownership. It is my (Prof Wittie's) current understanding that the copyright for any software you produce for a class belongs to you which means you are free to distribute/sell your software as you like. If your software is super awesome and you can actually make money off it, verify the copyright thing with Bucknell first and definitely tell me about it so I can brag about your awesomeness to future classes.

The following information will go into your report after the design questions

- What your software does (a sentence or two)
- Why it is appropriate to do this in your language (1 paragraph)
- A concise (**short**) user manual to let us know how to compile, run, and use your software (up to a page is fine)

You will also produce a video (1-5 min) of your software in action. You must speak in the video but you don't have to be seen in it unless you want to. In the video, tell me

- your name,
- your language's name,
- what your software does,
- show a demo of compiling it (if it won't/doesn't compile, just say so)
- show a demo of launching/running it, (if it won't run, just say so)
- tell me why this was (or wasn't in hindsight) a good idea in your language.

Hand in your video and your uncompiled code. I watch the videos and glance at your code. The software and accompanying video are worth 25% of the overall project.

Project Phases

These phases will keep you on track and help you complete your report. They are due by midnight as listed on the course schedule. I expect that all code you submit runs and demonstrates the features you claim for it.

The phases will be graded together as one homework assignment. Individual phases will likely get check minus, check, check plus style grades.

Your lecture instructor will let you know how many phases they expect you to turn in and when they are due.

Phase 0

Pick **three** of the following programming languages: Scala, Erlang, Ada, Perl, Javascript, TypeScript*, Dart*, Lua, Ruby, Php, Hack*, Eiffel, Go, Swift, C#, F#, Visual Basic, Objective-C, Fortran, Cobol, Rust, Lisp, OCaml, Elm, Mathematica, Clojure*, Mathlab, Racket*, Smalltalk *etc.*

You **must** be able to run the programs you write in your language. Languages marked with an aterisk (∗) above *may* not be available on our campus Linux, Mac, or Windows machines.

Other languages may be added to this list by request. (Generally not Java, Haskell, C, Python, Bash, Prolog, or C++). I will allow Latex but it is HARD to do for this project.

You need to submit the top **three** preferred programming languages you would like to study to **Moodle**. Your instructor will let you know which **one** language is yours.

(Grading based on having **three** languages on your list, handed in correctly, on time).

Phase 1

Write a helloworld program. It must include a comment which gives compilation and running instructions for Linux. It must also include a comment stating the language's name. You need a citation for this or a note saying you already knew how to write this code. See the section further down on proper citations.

Submit a test run with your program that shows you compiling/running it and what output it produced.

(Grading based on hello existing, having a comment as described above, a citation, and running properly).

Phase 2

What paradigm(s) is your language? The paradigm is the way in which you program in the language. It is supported by a set of features. What features of the language support your answer? Match specific features to specific paradigms. You need a citation(s) for where you got your info. See the section further down on proper citations. I don't expect to see any code for this phase.

For example,

Java is Object Oriented and Imperative. Its is Object Oriented because Java has classes which contain fields and methods. Java supports its classes with inheritance and access control (public, private, etc) and interfaces and abstract classes. Java is written in an imperative style (commands on what to do with memory and what operations to perform in what order) with loops and assignments (state).
All About Java, Not O'Really, pg 2-8

Be careful, just having a feature or two of a paradigm with no other support doesn't make something Object oriented or Functional. Python has objects and list comprehensions but is debatably not a functional or object oriented language.

Python is an Imperative Scripting language with some support for functional and object oriented features. Like most scripting languages, Python is interpreted and it is quick to program in. Python is written in an imperative style (commands on what to do with memory and what operations to perform in what order) with loops and assignments (state). Python has objects and inheritance but does not provide software engineering support for them such as inheritance and access control, interfaces and abstract classes. Python has list comprehensions and map and filter operations but does not support the all-function, state-free programming of the functional languages.
All About Python, Not O'Really, pg 2-8

Warning: Having functions does NOT make a language functional.

Don't bother mentioning the procedural paradigm (gives us functions and procedures) or block structured (gives us scopes) since pretty much all languages these days have that.

(Grading based on correctness, reasoning, and proper citations).

Phase 3

What software do you plan to implement? Why is it appropriate for your language? What features does it rely on? Give about a 5-10 sentence answer, absolute maximum one page. I don't expect to see any code and you may not need citations for this phase.

(Grading based on appropriateness of your choice and having a concise informative answer).

Code examples for design questions

Your answers need a code example or an argument to support them. Your code must actually compile and run. Show both the code and the test run. Provide instructions on how to compile and run your code. You must write all code yourself.

Bad code example:

```
print "hello";
```

This is bad because it's not a complete example. I don't know how to run this. I don't know what the code should do when run.

Bad code example (for the overall report):

```
See file hello.fz
```

This is bad simply because I want all the code examples in the overall report, not in separate files. This is just fine for the project phases assuming it also comes with the next example.

Good code example:

```
begin main()
   print "hello";
end
```

```
Compile and run: foozle hello.fz
```

```
Prints: hello
```

```
[1] (a citation)
```

You only have to explain how to run programs **once** in the full report. **The code for each phase needs to explain it each and every time so I can easily run the examples without looking up the instructions each time**. You should always provide code that can be run without further context (show the whole program) and you should say what you to expect to happen when it runs. A good program also only has code to demonstrate one thing. HelloWorld should not also show how to declare arrays unless it is needed to say hello.

Cite your references

You can use any books, knowledgeable websites (this means not random opinions found on the web unless you can show they were correct) or industry experts you find (Uncle Fred may be a language expert) for references.

Most languages have an official website and tutorial provided by the people who created the language. Your report must have references cited. (You can use footnotes or a reference section). I will want citations such that I can tell where you got each fact you looked up.

Websites need a name (The Foozle Manual) and the URL (www.foozle.com). If the website is many pages and you are just referencing one section, give the URL for that section (www.foozle.com/blah/examples/typeconstructors.html).

Books need a title, an author, the page number(s), and either the ISBN or the edition or the date published.

Personal communication needs a name, a note about why they are an expert or what they are an expert in (20+ years industry experience with Foozle), and the month/year you had the conversation.

Bad citation example:

fact
fact
fact

works cited:
1: www.website.com/qwerty/Foozle.html The official Foozle website
2: How to program in Foozle, Jane Doe, ...
3: Uncle Fred, 20 years industry exp, personal communication, Nov 2011
4: www.foozleexplained.com A Foozle tutorial, ...

These citations are bad because I cannot tell where you got which facts without reading each and every one of your cited references and guessing.

Good citation example:

fact [1]
fact [2,3]
fact [4]

works cited:
1: www.website.com/qwerty/Foozle.html The official Foozle website
2: How to program in Foozle, Jane Doe, ...
3: Uncle Fred, 20 years industry exp, personal communication, Nov 2011
4: www.foozleexplained.com A Foozle tutorial, ...

These citations are good because I can lookup more information on a topic and know which cited reference to use as a starting point.

Crowd source references

Wikipedia and stackoverflow.com are two sources that students commonly rely on too heavily. Wikipedia contains crowd sourced information and while the computer science portion of it is often factually correct, it is still crowd opinion and not a valid reference. Stackoverflow contains internet conversations on computer science topics and again is crowd opinion and not a valid reference. **Therefore, you may NOT directly use Wikipedia and Stackoverflow and sites like them.** You must verify any information found on these non-allowed sites by finding it in a knowledgable source such as the official website for your language or a published book on your language.

Using code written in references

You must write all code yourself but you are allowed to use **all** resources that aren't forbidden by the collaboration rules as long as you cite them in the assignment. This means you can **not** copy code from anywhere else. Here are some examples of how to **use** code found in references.

For example, your source contains the following code

```
x = 1:5      – lists
y = "a"*3    – repeated strings
z = x[2]     – index
```

Bad code usage example:

```
x = 1:5      – lists
y = "a"*3    – repeated strings
z = x[2]     – index
```

Compile and run: foozle list.fz

This demonstrates list functions[3].

This is copied code and is **not allowed**.

Good code usage example:

```
x = 3:8       – a range to make a list
y = “hi”*5    – repeated strings to make a list
z = x[0]      – index into a list (lowest index is 0)
print(x,y,z)
```

Compile and run: foozle list.fz

This demonstrates list functions [3].
Prints: [3,4,5,6,7,8] [“hi”,hi”,“hi”,“hi”,“hi”] 3

This demos the same concepts as your source but you used your own numbers and values and figured out what now prints (it prints different things than your source) which means you understand what is going on and it is now **your** example. You also wrote your own comments instead of the ones in the source example.

For example, your source contains the following code

```
object Main extends App {
val sqr = (x: Int) => x * x
val anonfun1 = new Function1[Int, Int] {
  def apply(x: Int): Int = x * x
}
assert(sqr(4) == anonfun1(4))
}
```

Bad code usage example:

```
object Main extends App {
val sqr = (x: Int) => x * x
val anonfun1 = new Function1[Int, Int] {
  def apply(x: Int): Int = x * x
}
assert(sqr(4) == anonfun1(4))
}
```

Compile and run: **foozle anonymous.fz**
This demonstrates anonymous functions. [3]
Prints: ...

Besides being unclear which parts of the above are actually the anonymous function, this is copied code and is **not allowed**. (Which part is important? The object Main? The sqr line? The anonfun1 stuff? The apply line? The assert line? This is a real example from a student project and only part of the sqr line was actually relevant).

Good code use example:

Assume you already had a hello world example (that you wrote)

```
def main() {
  print(3)
}
```

Here’s what you can do to **use** the above source of code to make your own example.

```
def main() {
  print(((x: Int) => x + 1)(1))
}
```

Compile and run: **foozle anonymous.fz**
Prints: 2
This demonstrates anonymous functions. [3]

The anyonymous function above computed a square of the input, your example adds one to the input. (Its not using anonymous for the same thing). The source gave a bunch of code that was not relevant to what you needed, your example only shows the relevant stuff. You still included a citation because it is where you learned how to write an anonymous function in Foozle.

Sample topics

Here are some sample topics for your report. We expect you to answer all the subquestions inside each one. Your answers must be supported by programs (embedded in the report) unless specified below. You can also have arguments supporting your answers. Don't forget to include references.

Do not use the ones that do not apply to your language or ones that have boring *no* answers. You can also make up your own topics/questions.

I've given *partial* answers for some of these as examples. I generally did not include the supporting code or arguments or references that you would need.

For all languages

- Primitive Types
  What types come with the language? (int? bool?) What kinds of values can they have? (0? -09? true? -00.30?) What range of values can they have? (0 to infinity? -2^{31} to 2^{31}?) How much memory does it each type use? (Bits and bytes - not all languages will let you know this.) Does the type grow to fit large values? (like Python's integer).

  For example in Java,

```
public static class Phase3 {
  public static void main(String args[]) {
     bool a = true; // 1 bit booleans (true, false)
     char b = 'x'; // 16 bit character
     int c = 8; // 32 bit integer -2^31 + 1 to 2^31 + 1
     byte d = 3; // 8 bit integer
     int e = 05; // octal (base 8)
     int f = 0x1234; // hexidecimal (base 16)
     short g = 5; // 16 bit integer
     ...
  }
}
```

  Compile (javac Phase3.java) and run (java Phase3).
  Nothing prints.
  The types are fixed sizes (don't grow).
  All About Java, O'Really, pgs 2-8

  For example in C:

```
#include <stdio.h>
int main() {
   int c = 8; // integer -2^31 + 1 to 2^31 + 1
   printf("%d",sizeof(c)); // prints the size in bytes
  ...
}
```

  Compile (gcc Phase3.c) and run (a.out).
  On a 64-bit machine running Linux and using gcc version 4.4.7 20120313 (Red Hat 4.4.7-3) (GCC)
  this prints 4 so it uses a 4 byte integer.
  K&R *a citation* says that the size of the primitives depends on the specific machine, operating system, and compiler.

- Composite and Constructed Types
  (tuple, class, struct, matrix, enum...) What formal definitions of constructors from your book and lecture do they each

match? (record, cartesian product, array, union, enum, ...) If the type constructor has options, what choices did your language make? Give an example program creating and using each of these type constructors.

For example in C

An array (an array or mapping in lecture)

```
#include <stdio.h>
int main() {
   int x[10]; // declaring an array
   x[0] = 5; // setting a value
```

K&R *citation*

Indexes are integers from 0 to size-1 (inclusive). It can contain any type but all contents are the same type. Array length is fixed once declared. You cannot access the length. No extra functions come it and you can't add any either. You can make arrays be multidimentional *and showing how is a separate phase*.

- Background
  Give brief background of the language and its current state. This question does not require additional code examples or arguments to back up your reasoning.
  - When was it created (range of years or a year)
  - Who created it (person, committee, group at MIT, ...)?
  - Is it compiled or interpreted? (Or can you do both with it?) If it's somewhere in between, say so and explain.
  - Was it created for a purpose? What purpose?
  - Is anybody hiring programmers for this language right now? Who? What kind of jobs?
- Default parameters
  Can functions have default parameters like the ones that Python permits?

```
def foo(x, y=3, z="hi"):
   print(z)
   return x+y
```

- User defined types (beyond the type constructors)
  These generally make type aliases. Some languages let you specify the number of bits in an integer.

  For example, typedef or #define in C.

```
typdef foof int
foof x = 3;
```

- Static vs Dynamic typing
  (Do not answer this if your language is untyped)

  Does the variable or the value hold the type? What is the default? Can you make the other option happen too? What type checking is done either at compilation time or at the way beginning and what is done as it runs?

  In Java, statically typed, it does almost everything during compilation except things like array bounds checks and divide by zero checks and non existant file checks.

- Implicitly or explicitly typed
  (Do not answer this for dynamically typed languages unless yours is explicitly typed)

  Be careful, implictly typed does not mean untyped or dynamically typed. Haskell, strongly typed and statically typed, can choose to be implictly typed. Its types can be inferred.

  For example, in C all declarations are explicitly typed (globals and parameters and local variables are declared with types, struct fields are declared with types, even functions have explicit return types).

- Nominal vs Structural typing
  What is the default? Can you make the other option happen too?

- Typing Strength
  untyped vs weakly typed vs halfway tween weak and strong vs strongly typed (This means you do a lot of implicit coersion tests and look for lossy or stupid coersions).

- Reflection
  Computational and/or structural?

  For example, Java has some reflection capabilities (foo.class()) and Smalltalk has a ton of reflection both computational and structural. As far as I know, Haskell has none at all.

- Memory Hazards
  which ones can you make? Which are not possible? Why? Do not answer this if no hazards are possible.

  For example, C has aliasing, dangling references, and memory leaks. Java has only aliasing.

- Static vs Dynamic Scope
  What is the default? Can you make the other option happen too? For example, Java and C are statically scoped.

  (Don't answer this question if every variable is global so your language is unscoped unless you have built in ways to do static/dynamic anyway)

- Polymorphism
  Adhoc? Inclusion? Parametric? Give examples of all the ones it can do.

- Overloading of function or method names or operators
  Can you overload existing operator symbols? Can you overload non-existant operator symbols? If you have several functions with the same name in one scope does it overload and allow them all or override and allow only the last (or error and take none)?

  For example, Java allows overloaded method names in a class. C++ allows overloaded existing operators. Python uses the last one you wrote and ignores the previous ones which means it doesnt do function overloading but its still interesting.

  (Yup, this question might also give you an answer for adhoc polymorphism)

  (Do not answer this question (or any question) if the answer is just no).

- Strict vs Non-strict Evaluation
  What is the default? Can you make the other option happen too?

- Parameter Passing/Calling Mechanisms.
  For strictly evaluated languages: Pass by value or reference. For non-strictly evaluated languages: Call by name or need. What is the default? Can you make the other option happen too? Are some parts of the language the opposite of the others?

  For example, C passes integers by value but arrays by reference. Be wary of references.. Java passes everything by value (its objects are all references that get passed by value).

- The string type
  Is it a primitive type or added in likely by a library? Is it made with one of the type constructors? What math operations can you do on strings? (s+s, s*3) What other operations are provided for strings? (none, substring, index of). Is there a concept of a terminating character like the '\0' in C?

- Math operations on numbers
  Only answer this if there are non-standard math operations (not $+$ $-$ $*$ / mod power). Can you directly access or manipulate the bits of an integer variable?

  For example, You can in C, you cannot in Java. If you can, what operations are possible? $<<$ in C does a bit shift.

- Multi dimentional arrays
  Are there limits to how many dimentions they can have? How do you make one? How do you access the fields?

  For example, C and Java both allow this. Java even has some support for it. Can they be jagged (each row is a different length)?

- Variables
  What kinds of variables are there? (locals, globals, class variables, static variables, constants, etc). Give code examples of each.

- Dangling Else
  Is it a problem? If yes, how does it handle it? If not, why is it not a problem? It is a problem in both C and Java but not in Haskell. **You can answer this even if the answer is no but you must show code to prove it either way.**

- Coersion
  Does it do implicit type conversions? Which ones? Does it allow explicit conversions? Which ones? A chart and some examples (possibly not every example) would answer this.

  For example, in C, int x = 3.4; has an implicit conversion. int x = (int) 3.4; has an explicit conversion.

  (Yes, you can answer both this and the typing strength question counting as two separate questions)

- Garbage Collecting
  Does it have a garbage collector? What does it do? Can you trigger it yourself? Mark sweep or reference counting at runtime? Does it insertion manual collection code for you during compile time? If it has programmer-written manual collection, give examples. This may not need code if its done automatically.

  For example, Java uses a garbage collector which you can trigger if you really want to (an example of triggering it would be nice.)

- Short Circuit Evaluation
  Does it use short circuit evaluation? Are there operators with options to short circuit or not as you like? **You can answer this even if the answer is no.** Show tests for both *and* and *or*.

  For example, C uses short circuit on the && and || operators. Python has *and* and *or* and also symbols so you can choose to short circuit or not as you like.

  (Be wary, & and | are usually bit-wise math operators not logical or and and operators. If so, don't use them in a short circuit test).

- What is a BNF grammar for it? Show **only** the BNF for a common and recognizable portion of the language such as a loop or an if statement or a function call. This does not need code.

- Fix of operators
  Does it use any non-standard infix, prefix, postfix, mixfix operators? function calls? some combo of them?

  For example, Java uses infix binary operators (a+b) and some pre/postfix unary operators (++i, i++), and prefix function calls foo(a,b,c). A Java report won't use this question because the answer is standard.

- Control structures
  What non-standard control structures does it use? (For loops, while loops, do while loops, and if statements are standard in an imperative language. If expressions are standard in a functional language. Return is standard pretty much everywhere. The usage of main as a starting function for imperative code, and code outside any function as a starting point for scripting and functional languages is standard. Iterator for loops are becoming standard in all imperative languages.) What stuff transfers control around the code so that it doesnt just run the first line and then the second line then the third etc.

  For instance C has an old style ?: for selection, If, For, Return, Function calls, main function, loops, etc. The ?: is the only non-standard one in the whole collection.

- Language pre-processors and macros
  Like the `#include` and `#ifndef` of C.

- Portable
  Does it claim to be portable like Java? (Carefully define portable as Java uses the term. http://en.wikipedia.org/wiki/Cross-platform) How portable is it? This does not require code.

- Tokens
  What token categories does it have? (keywords, literals, symbols, identifiers) What are the regular expressions for its tokens? (Give a regular expression to describe each category of literals - integers, booleans, etc). Does it have a token delimiter? (What separates the tokens? spaces?) If so, what? Are the keywords reserved? (If reserved then you can not name your int *if*). This question does not need code.

- Free or fixed format?
  Dependent on the specific compiler? Always that way?

Object Oriented Features

- Does it have exceptions? If not, then how does it do error handling? Does it have something equivalent to Java's finally? Can a try block have mutiple catch blocks? Can you create your own exceptions or just throw existing ones?

- How inheritance works
  Access controls? Multiple inheritance? How do constructors and destructors work with inheritance? What kinds of variables/constants get inherited in what ways (public/private inheritance does what?) C++ lets you choose what access control stuff will have after it is inherited. Java says its all the same access control as the parent had. You only inherit the public stuff in many languages. Do all objects automatically inherit from one base class (in C they do not, in Java they all inherit from Object)?

- Static vs Dynamic method binding
  What is the default? Can you make the other option happen too?

- Deep vs Shallow copying of objects
  What is the default? Can you make the other option happen too?

- Namespaces
  Does it have this concept?

- Constructors/Destructors
  How does it do constructors? Default constructors? Do you still get a default constructor if you write one that takes a parameter (non-default)? Can you have multiple constructors (do you overload the function names to do it)? Do you write destructors? Is there a default destructor? Can you have multiple destructors?

- Abstract classes

- Interfaces

- Friend classes

- (OOPL only) Other Object oriented feature
  like super (C++, Java, etc). You can use this question once per feature.

- (NON OOPL only) Borrowed Object oriented feature
  Python has classes but almost no support. You can use this question once per feature.

Functional Features

- Higher order or first class functions (This is one question, not two)
  A first class function can be assigned to a variable. A function is higher order if it takes or returns functions.

  Haskell supports these in many ways. C lets you pass functions into a function which is higher order but thats it. Map, fold, filter, composition, etc.

  Be careful foo(bar()) is likely NOT demonstrating higher order. Its probably just calling both foo and bar.

- Function composition using some sort of composition operator. We will cover this is Haskell.

- Infinite/Lazy Lists
  Haskell supports these.

- Anonymous Functions
  Can you call them on an argument without storing them in a named variable? Can you store them in a variable? (first class) Can you pass them to other functions as uncalled parameters (higher order)?

- Pattern Matching (built into the language) (Don't choose this question if you do it using if/else and isType() statements)

- Let bindings

- List comprehensions (built into the language) (Don't use this question to say you can call map and filter - use those for the higher order functions question)

- (Functional only) Other Functional feature
  closures, continuations, etc. You can use this question once per feature.

- (NON Functional only) Borrowed Functional feature
  Python has list comprehensions. You can use this question once per feature.

Scripting Features

- Regular expression support
  Is it built in or done with a library? Does it do match (whole thing)? Search (find inside)? Is search greedy or non greedy? Can it do find all (finds all non overlapping searches inside)? Does it do capture (aka grouping in Python)? Does it do replace (aka substitution)?

  Java has a regular expression library (so does Python).

- (Scripting only) Other Scripting feature You can use this question once per feature.

- (NON Scripting only) Borrowed Scripting feature
  You can use this question once per feature.

Imperative Features

- Statement vs Expression and Function vs Procedure
  What stuff is a statement? What stuff is an expression? What is both? Do you have to label functions differently from procedures? Can you have both functions and procedures?

- Pointers/references
  C allows references to anything. Java requires references to Objects and Arrays. Can you access them as references?

  Java lets you manipulate them.

```
int[] arr = new int[3];
int[] arr2 = arr; // you set a reference directly
```

  Can you manipulate them directly? ($\&x$ or $\&x+3$ in C - look up "pointer math" to see examples).

- Stack vs Heap
  what goes where? In Java, arrays are all on the heap. primitive contents are actually inside the array. object contents are somewhere else on the heap. In C, array are where ever you want them. Contents are in the array or out on the heap as you like.
- (Imperative only) Other Imperative feature
  You can use this question once per feature.
- (NON Imperative only) Borrowed Imperative feature
  Some functional languages allow state. You can use this question once per feature.

Logic Features

- Backtracing. We cover this in Prolog.
- Unification. We cover this in Prolog.
- Resolution. We cover this in Prolog.
- (Logic only) Other Logic feature
  You can use this question once per feature.
- (NON Logic only) Borrowed Logic feature
  You can use this question once per feature.