There are two programming languages that I have seen that are new to me, PHP and R. They both target different audiences. I will begin by talking about R.

R is a programming language that started development in 1991 and was released to the public in 2000. It was developed by two main programmers, Ross Ihaka and Robert Gentleman. They created R as a tool for working with statistics and graphing math data. It seems to be a very niche programming language that seems to target people who are into mathematics. The language uses expressions a lot like Ocaml. I think one interesting thing about the language is the mention of objects in it. In the manual, it says variables, arrays of numbers, character strings, and functions are objects. These objects can be stored permanently from an R session in a file, which I think is interesting. When I have learned of other programming languages like Java, C, Ruby, Ocaml, or Rust, we never really thought of saving an instance of data (object). We thought more so of saving the final result of that data, but not necessarily an instance of an object itself. With that being one of the first few things mentioned in the manual, I find that interesting. Some of the syntax kind of reminds me a bit of what we covered in lecture last week. When assigning a value to a variable, you use a function. One example is this

```
> x <- c (10, 2, 5, 3)
```

Another way to do that same assignment is

```
> assign(“x”, c(10, 2, 5, 3))
```

Both of these assigns the value of the array to the variable x by a function.

R seems to be a very functional programming language, which makes sense since it is supposed to be geared towards people who are into math. Math is just a bunch of expressions that evaluate to a value, so that programming paradigm for the language makes sense. R has lists, which are defined as recursive structures in the language, which reminds me of Ocaml. A list seems to be like a set of pointers in R to other objects that could themselves be a sort of “list”. For example

```
> nums ← c(5,4,3,2)
> words ← c(“hello”, “world”)
> elist = list(nums, words)
```

elist is a list that contains the objects nums and words. We create this list by using a function called list. This list has different data types, which makes me think that R is a dynamically typed language. So that’s one cool feature of R that reminds me of the typing system we learned about with Racket.

As far as I can tell, R is still used today. I have heard other people still use R in academia today. When I took stats 400 last semester, my TA told me she did some programming in R. I’ve heard of other students also using R as well in their classes. Outside of class, I’m not sure how well used it is. I would assume since it’s used in a university setting it would also be used outside of academia as well. This language still seems relevant today in academia.

https://cran.r-project.org/doc/manuals/r-release/R-intro.html
https://www.r-project.org/about.html
https://bookdown.org/rdpeng/rprogdatascience/history-and-overview-of-r.html

The other programming language that I have seen that is new to me is PHP. This language was developed by Rasmus Lerdorf in 1994. It was originally used to keep track of how many people visited Rasmus' online resume. PHP also did not originally start out as a programming languages. Rasmus added more features so that PHP could help websites interact with databases so users could develop dynamic web applications. He slowly added more features to it as well, such as the ability to have built in database management, user defined function support, and object-oriented programming support over the years. This took place between 1994 to 1998.

Rasmus added more features to PHP to create a programming language that would help in the creation of dynamic web pages. It was targeted mostly to web developers and as a software to be installed on hosting servers for websites. PHP was developed to make it easier for web developers to create dynamic web pages by having a language that interacts with a lot of features needed for a dynamic web page, such as databases and DOM manipulation.

```
<?php

$makefoo = true;

/* We can't call foo() from here
since it doesn't exist yet,
but we can call bar() */

bar();

if ($makefoo) {
function foo()
{
echo "I don't exist until program execution reaches me.\n";
}
}

/* Now we can safely call foo()
since $makefoo evaluated to true */

if ($makefoo) foo();

function bar()
{
echo "I exist immediately upon program start.\n";
}

?>
```

This code, to me, is interesting. It tells us a few different things. One thing that it tells us is that code is read from top to bottom. One weird thing that I notice is the calling of the functions bar() and foo(). Bar() is declared after it is called. This is ok and it will not result in any error. However, when we have the condition $makefoo set to true, and that evaluates to true, we make the function foo(). We can then call that function foo() after the program is done running. We cannot call foo() before it is made in the if condition, but we can call it after we evaluate the first if condition, that is, if the first one evaluated to true. This makes me think that PHP evaluates things in a certain order. Maybe it evaluates the values that if conditions generate AFTER the value of other things like variables and functions. In the PHP manual, we see it say that almost everything in PHP is an expression that evaluates to a value, even functions. So I think that is very interesting that, if we had called the function foo() before it was made from the first if condition, it would not be called properly. However, after we go through the first if conditional, then it is safe to call foo() in the subsequent if condition. But what if the first one evaluated to false? We would not have that function foo() and we could still not call that function safely, and possibly have an error.

It seems that PHP is still used today on plenty of web servers. Based on the graphs seen on w3techs.com, it seems that more web servers use PHP than any other similar software. So I would say that PHP is still used today and it still seems to be very relevant.

https://www.php.net/manual/en/history.php.php
https://www.php.net/manual/en/language.expressions.php
https://www.php.net/manual/en/control-structures.if.php
https://www.php.net/manual/en/functions.user-defined.php
https://w3techs.com/technologies/details/pl-php