R PROGRAMMING LANGUAGE

Members:
Abreu, Luke
Emeterio, Ricamelle
Lansangan, Patrishia
Sabado, Patrick John
Zamora, Vinalyn

Class Code: 9354A
10:00-11:00 TF D515

I. Introduction

R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files. R is freely available under the GNU General Public License and highly extensible meaning it can be extended or expanded from its initial state that might support plug-ins that can add extra functionality to the program through the use of user-submitted packages for specific functions and graphics that is supported by the R Foundation for Statistical Computing.

II. Historical Perspective

R is a dialect of S developed by John Chambers at Bell Labs. In 1996 S language was initiated as an internal statistical analysis environment originally implemented as FORTRAN libraries, but in early versions of the language did not contains functions for statistical modeling.

R was created in 1991 by two professors at the University of Auckland in New Zealand Ross Ihaka and Robert Gentleman. In 1993 first announcement of R to the public. In 1995 Martin Mächler convinces Ross and Robert to use the GNU General Public License to make R a free software and in 1997 R core team was developed, so they can modify the R source code.

The original creator of S Language, John Chambers, is now part of the R Development Core Team and this programming language was named R, based on the first letter of first name of the two authors Ross and Robert.

III. Language Design Goals and Language Paradigms

A. Language Design goal

- For computationally intensive tasks, C, C++, and FORTRAN code can be linked and called at run time.
- Advanced users can write C, C++, Java, .NET or Python code to manipulate R objects directly.
- R is highly extensible through the use of user-submitted packages for specific functions or specific areas of study.
- It is designed to be an object-oriented programming and functional programming structure.
- It is designed to be portable and accessible to the other platform because R language is interpreter.

B. Language Paradigms

- Array Programming
- Object-Oriented Programming
- Imperative Programming
- Functional Programming
- Procedural Programming
- Reflective Programming

IV. Language Application Areas

- R Programming applications compass the universe from hypothetical, computational statistics and the hard sciences, for example, astronomy, chemistry, and genomics to

practical applications in business, drug advancement, finance, health care, marketing, medicine and much more.

V. Language Environments, Tools, Libraries, Frameworks, and Documentation

A. Language Environments

- Windows
  - R is available as a source code for many versions of Windows
- Mac
  - R Installers are available in R-patched and R-devel.
- Linux
- R is available as a binary code for many versions of Linux.

B. Language Tools

A. Open Source IDE's

- StatET
- R Analytic Flow
- RCommander
- R Studio
- JASP

B. Source Code Editor

- EditPad Pro
- Vim
- Aquamacs
- Crimson Editor
- Syn

C. Test Tools

- R Unit Test Framework
- Testthat
- SVUnit

C. Language Libraries

In R, there were over 11,000 packages available on the Comprehensive R Archive Network, or CRAN. This huge variety of packages is one of the reasons that R is so successful. You also have bioconductor which has packages for the analysis of high-throughput genomic data.

Other R package resources like R-Forge, a central platform for the collaborative development of R packages, R-related software, and projects. R-Forge also hosts many unpublished beta packages, and development versions of CRAN packages.

D. Language Frameworks

1. Fselector
2. Mlr
3. Rgp
4. Text2vec
5. Tm

VI. Main Language Features and Constructs

A. Main Language Features

- Data sets are saved between sessions, so you don't have to reload each time.
- In addition to enabling statistical operations, it's a general programming language, so that you can automate your analyses and create new functions.
- A public-domain implementation of the widely-regarded S statistical language. R or S is the de facto standard among professional statisticians.
- Potentially much faster execution speed, less debugging since you write less code and easier transition to parallel programming.

B. Constructors

Arithmetic Operators	
Operator	**Description**
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponent
%%	Modulus
%/%	Integer Division

Assignment Operators	
Operator	**Description**
<-, <<-, =	Leftwards assignment
->, ->>	Rightwards assignment

Atomic Classes	
Type	**Example**
Character (Unicode)	"a","swc"
Numeric (real or decimal)	2, 15.4
Integer	2L
Logical	TRUE, FALSE
Complex	7i

Logical Operators	
Operator	**Description**
!	Logical NOT
&	Element-wise logical AND
&&	Logical AND
|	Element-wise logical OR
||	Logical OR

Relational Operators	
Operator	**Description**
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to

Reserved Words				
if	else	repeat	while	function
for	in	next	break	TRUE
FALSE	NULL	Inf	NaN	NA
NA_integer_	NA_real_	NA_complex_	NA_character	

Operator Precedence		
Operator	**Description**	**Associativity**
^	Exponent	Right to Left
-x, +x	Unary minus, Unary plus	Left to Right
%%	Modulus	Left to Right
*, /	Multiplication, Division	Left to Right
+, -	Addition, Subtraction	Left to Right
<, >, <+, >=, ==, !=	Comparisons	Left to Right
!	Logical NOT	Left to Right
&, &&	Logical AND	Left to Right
|, ||	Logical OR	Left to Right
->, ->>	Rightward assignment	Left to Right
<-, <<-	Leftward assignment	Right to Left
=	Leftward assignment	Right to Left

- **Control Structures**
  - **If**

```
if (test_expression) {
  statement1
}
```

  - **For**

```
for (val in sequence){
   statement
}
```

  - **While**

```
while (test_expression){
  Statement
}
```

- **Functions**

```
func_name <- function (argument) {
 statement
}
```

- **Return**

```
return (expression)
```

- **Switch**

```
switch (statement, list)
```

- **Data Structures**
  - **Vectors**
    c () function
    x <- c (1, 5, 4, 9, 0)

  - **Matrix**
    matrix () function
    matrix (1:9, nrow = 3, ncol = 3)

  - **List**
    list () function
    x <- list ("a" = 2.5, "b" = TRUE, "c" = 1:3)

VII. Sample Program Codes

Sample Code 1

```
print("Hello World!")

print("Hello World!", quote = FALSE)

print(paste("How","are","you?"))
```

Sample Code 2

```
num = as.integer(readline(prompt="Enter a number: "))

factorial = 1

if (num < 0) {

   print ("Sorry, factorial does not exist for negative numbers")

} else if (num == 0) {

   Print ("The factorial of 0 is 1")

} else {

   For (i in 1:num) {

      factorial = factorial * i

   }

   print (paste("The factorial of", num ,"is",factorial))

}
```

Sample Code 3

```
add <- function(x, y) {
        return(x + y)
}

subtract <- function(x, y) {
         return(x - y)
}

multiply <- function(x, y) {
          return(x * y)
}

divide <- function(x, y) {
          return(x / y)
}

print("Select operation.")
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")

choice = as.integer(readline(prompt="Enter choice[1/2/3/4]: "))

num1 = as.integer(readline(prompt="Enter first number: "))
num2 = as.integer(readline(prompt="Enter second number: "))

operator <- switch(choice,"+","-","*","/")
result <- switch(choice, add(num1, num2), subtract(num1, num2), multiply(num1,
        num2), divide(num1, num2))

print(paste(num1, operator, num2, "=", result))
```

References:

R (programming language). (2017, September 7). Retrieved from https://en.wikipedia.org/wiki/R_(programming_language)

Peng et al. (n.d.). Lecture 7 - Overview and History of R. Retrieved from https://www.coursera.org/learn/r-programming/lecture/pAbaE/overview-and-history-of-r

Vaishnavi Agrawal (2016, February 25). Applications of R Programming in r-eal World - eLearning industry. Retrieved from https://elearningindustry.com/applications-r-programming-r-eal-world

Wickham et al. (n.d.). Introduction to R. Retrieved from https://ramnathv.github.io/pycon2014-r/

ThienSi Le (2016, January 13). Statistical & Programming features of R. Retrieved from https://www.linkedin.com/pulse/statistical-programming-features-r-thiensi-le

Burns Statistics (2002). Why use the R Language. Retrieved from http://www.burns-stat.com/documents/tutorials/why-use-the-r-language/

Hadley Wickham (n.d.). Introduction. Retrieved from http://r-pkgs.had.co.nz/intro.html

Programiz (2011). Learn R Programming. Retrieved from https://www.programiz.com/r-programming