



- User Guide -

Diana-Alexandra Crintea – dac1u17
Filip-Constantin Andonie – fa2u17

COMP2212

Programming Language Concepts
Pair Project

❖What is SpellBook?

Functionality. SpellBook is a domain-specific programming language that performs certain simple computations on potentially *unbounded integer sequences* - **streams**. The streams are sequences of data, common in mathematics, computer science and everyday life. Examples of streams include streaming a movie on the web, a sequence of bits flowing through a wire in a digital circuit.

SpellBook is a sequential-type programming language, hence, if you are already coding in Pascal, C++ or Python, you will find that SpellBook programming language has a familiar built architecture. If you are a Harry Potter fan, great news! This programming language uses Harry Potter magical spells as commands!

Data Handling. This programming language expects as input a file containing positive or negative integers.

The data is read from the file as a list of streams, each stream being read *column by column* and saved into the Environment as a variable with the name **horcrux_n**, where **n** represents the index of each **stream**.

E.g. 1 3
 2 2 will memorise the sequences [1, 2, 3] and [3, 2, 1]
 3 1

Error Handling. This programming language detects common errors and fetches additional information and fixing suggestions for the ease of the user. Some of the possible errors are:

- **Morsmordre! Perhaps you forgot to FinitIncantatem your program.** - parsing error indicating that there might be an end statement missing
- **Morsmordre! There is an error on line:column! Mismatched brackets!** - mismatched brackets on the specified *line:column*
- **Morsmordre! There is an error on line:column! Perhaps you forgot to declare a conditional statement within the WingardiumLeviosa spell.** - missing conditional statement in 'while' type loop
- **Morsmordre! There is an error on line:column! Perhaps you forgot to declare a conditional statement within the Confundo spell.** - missing conditional statement in 'if' statement
- **Morsmordre! There is an error on line:column! Perhaps your Incendio spells are incomplete.** - missing body for 'if then' statement
- **Morsmordre! There is an error on line:column! Perhaps you forgot to make some spells inside Alohomora-FinitIncantatem.** - empty body
- **Morsmordre! There is an error on line:column! You forgot to close the list!** - missing ']' at the end of declaring a list
- **Morsmordre! There is an error on line:column! Your list cannot have null elements!** - the input file contains illegal input streams
- **Morsmordre! There is an error on line:column! Perhaps you forgot to write one of the following:**
 - Imperio in WingardiumLeviosa spell** - missing 'do' in 'while..do' loop
 - Incendio or Aguamenti in Confundo spell** - missing 'then' or 'else' in 'if then else' statement
 - Vestigium in Appare spell** - missing 'in' in 'let..in' statement
- **Morsmordre! Invalid input file! Non-Integer found!** - the input file contains illegal characters (not numbers)
- **Morsmordre! Invalid input file! There cannot be streams of different lengths!** - the input file contains streams of different lengths; all streams should have the same length
- **Expecto Patronum! Type mismatched! Integer Expected!** - found Boolean or List instead of Integer
- **Expecto Patronum! Type mismatched! Boolean Expected!** - found Integer or List instead of Boolean
- **Expecto Patronum! Type mismatched! List of Integers Expected!** - found Integer or Boolean instead of List
- **Riddikulus! Variable x not in scope!** - there is not any value assigned to 'x'
- **Baubillious! Index out of bounds!** - trying to fetch elements from outside of the bounds of a list

- **Avada Kedavra! Type mismatched in Episkey spell! You cannot mix spells and charms!** - type mismatched within '=' operation
- **Avada Kedavra! Type mismatched in Impedimenta spell! You cannot mix spells and charms!** - type mismatched within '!=' operation

❖ Using SpellBook Programming Language

Data types. SpellBook supports operations with Integers, Booleans and Lists of Integers.

As constants, there are: **lumos** - equivalent for the Boolean *true*
nox - equivalent for the Boolean *false*

A list of integers is being declared within squared brackets - '[' ']' and with a comma between elements.
 E.g. [1,2,3,4]

Syntax. The whole body of the code is wrapped between **Alohomora** (*begin*) and **FiniteIncantatem** (*end*) statements, as well as the bodies of all statements. The following tables describe the syntax of all statements and expressions.

Round brackets - '(' ')' are used only for tidying the code. As well as whitespaces, they do not change the meaning of the code

Statement	Meaning / Pseudocode
Fidelius <var> <expr>	Assignment statement: <var> = <expr>
Appare Fidelius <var> <expr> Vestigium <body>	Let <var> = <expr> in <body>
Confundo <expr> Incendio <body>	If <expr> then <body>
Confundo <expr> Incendio <body> Aguamenti <body>	If <expr> then <body> else <body>
WingardiumLeviosa <expr> Imperio <body>	While <expr> do <body>
Flagrate <expr>	Print <expr> in stdout

Expressions	Input	Output	Description
Legilimens x	x ← Integer	[Integer]	Returns the n th stream from the input file
Engorgio x y	x ← Integer; y ← Integer	Integer	x + y
Reducio x y	x ← Integer; y ← Integer	Integer	x - y
Geminio x y	x ← Integer; y ← Integer	Integer	x * y
Diminuando x y	x ← Integer; y ← Integer	Integer	x / y (x div y)
Caterwauling x y	x ← Integer; y ← Integer	Integer	x % y (x mod y)
AlarteAscendere x y	x ← Integer; y ← Integer	Integer	X ^y (x to the power of y)
Entomorphis x y	x ← Integer; y ← Integer	Boolean	x < y
CarpeRetractum x y	x ← Integer; y ← Integer	Boolean	x ≤ y
Defodio x y	x ← Integer; y ← Integer	Boolean	x > y
Deprimo x y	x ← Integer; y ← Integer	Boolean	x ≥ y

Episkey x y	x ← Integer; y ← Integer	Boolean	x == y (x equal to y)
	x ← [Integer]; y ← [Integer]		
	x ← Boolean; y ← Boolean		
Impedimenta x y	x ← Integer; y ← Integer	Boolean	x != y (x not equal to y)
	x ← [Integer]; y ← [Integer]		
	x ← Boolean; y ← Boolean		
Crucio x	x ← Boolean	Boolean	!x (not x)
Serpensortia x y	x ← Boolean; y ← Boolean	Boolean	x && y (x and y)
Evanesce x y	x ← Boolean; y ← Boolean	Boolean	x y (x or y)
Accio n x	n ← Integer; x ← [Integer]	Integer	Returns the n^{th} element of x
Confringo a b x	x ← [Integer]	[Integer]	Returns the list of elements of x from index a to index b
Ascendio x	x ← [Integer]	Integer	Returns the first element of x
PrioriIncantatem x	x ← [Integer]	Integer	Returns the last element of x
Informous x	x ← [Integer]	Integer	Returns the length of x
Ferula x	x ← [Integer]	Integer	Returns the sum of all elements of x
Depulso n x	n ← Integer; x ← [Integer]	[Integer]	Adds n to the end of x and returns the list obtained
Flipendo n x	n ← Integer; x ← [Integer]	[Integer]	Adds n to the beginning of x and returns the list obtained
Expelliarmus n x	n ← Integer; x ← [Integer]	[Integer]	Removes from x the n^{th} element and returns the list obtained
Ventus x	x ← [Integer]	[Integer]	Removes the first element from x and returns the list obtained
Obliviate x	x ← [Integer]	[Integer]	Removes the last element from x and returns the list obtained
EverteStatum x	x ← [Integer]	[Integer]	Returns the list of elements from x in reverse order
Epoximise x y	x ← [Integer]; y ← [Integer]	[Integer]	Returns the list obtained from the concatenation of x and y
Pack	-	Integer	Returns the size of the environment (the number of variables used) If utilised before initialising a variable returns the number of input streams

Comments. SpellBook allows three types of comments:

- Single line comment

Illegibilus <comment>

- Multi-line comment

Illegibilus <multi-line comment> **MischiefManaged**

- Author's comment

I <name> **solemnly swear that I am up to no good**

We <names> **solemnly swear that we are up to no good**

❖ Appendix

Colour scheme:

	- comment
	- author's name
	- begin/end statement
	- expression that returns an Integer
	- expression that returns a List
	- expression that returns a Boolean
	- statement
	- constants
	- variables, values and brackets

Note! Some of these problems do not apply the most straight-forward algorithms, as we wanted to emphasise more on functionality.

Problem 1 - Prefixing

Take a sequence $a_1 a_2 a_3 a_4 a_5 \dots$ as an input and output the sequence $0 a_1 a_2 a_3 \dots$, that is, the sequence that is the same as the input sequence, but starting with a single 0 character.

pr1.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Illegibilus the next statement stores in variable **x** the first stream from the file

Fidelius **x** **Legilimens** **0**

Illegibilus the next statement prints **(0 : (init x))** - drops the last element in **x** and adds **0** to the beginning

Flagrate Flipendo **0** **Ventus** **x**

FiniteIncantatem

Problem 2 - Copying

Take a sequence $a_1 a_2 a_3 a_4 a_5 \dots$ as an input and output two copies of it

pr2.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Illegibilus the next statement stores in variable x the first stream from the file

Fidelius x **Legilimens** 0

Illegibilus the next two statements print x , such that the program outputs two copies of x

Flagrate **Legilimens** 0

Flagrate **Legilimens** 0

FiniteIncantatem

Problem 3 - Stream arithmetic

Take two sequences $a_1 a_2 a_3 a_4 \dots$ and $b_1 b_2 b_3 b_4 \dots$, and produce the sequence $a_1 + 3b_1$
 $a_2 + 3b_2 a_3 + 3b_3 a_4 + 3b_4 \dots$

pr3.spl

We Diana and Filip solemnly swear that we are up to no good

Alohomora

Illegibilus the next statement stores the value 0 in variable i used as an index

Fidelius i 0

Illegibilus the next two statements store the length of the first stream from the input file in variable k and the second one in l

Fidelius k (**Informous** **Legilimens** 0)

Fidelius l (**Informous** **Legilimens** 1)

Illegibilus the next statement initialises `out` with an empty list

Fidelius `out []`

Illegibilus if the lengths of the two streams are equal, thus they both exist in the environment

Confundo **Serpensortia** (**Defodio** k 0) (**Defodio** l 0)

Illegibilus then while $i < k$

Incendio **Alohomora**

WingardiumLeviosa **Entomorphis** i k **Imperio**

Alohomora

Illegibilus $s = \text{get}[i] \text{ fstStream} + 3 * \text{get}[i] \text{ sndStream}$

Fidelius s **Engorgio** (**Accio** i **Legilimens** 0) (**Geminio** 3 **Accio** i **Legilimens** 1)

Illegibilus add s to the end of out

Depulso s out

Illegibilus $i=i+1$

Fidelius i **Engorgio** i 1

FiniteIncantatem

Illegibilus print out

Flagrate out

FiniteIncantatem

Illegibilus else print the first stream

Aguamenti **Alohomora**

Flagrate **Legilimens** 0

FiniteIncantatem

FiniteIncantatem

Problem 4 - Accumulator

Take a sequence $a_1 a_2 a_3 a_4 \dots$ and output the sequence $a_1 a_1+a_2 a_1+a_2+a_3 a_1+a_2+a_3+a_4 \dots$, where each term of the output is the sum of all the input terms up to that point.

pr4.spl

We **Diana** and **Filip** solemnly swear that we are up to no good

Alohomora

Illegibilus the next statement stores the value 0 in variable i used as an index

Fidelius i 0

Illegibilus the next statement store the length of the first stream from the input file in variable k

Fidelius k (**Informous** **Legilimens** 0)

Illegibilus the next two statements initialise x with 0 and out with an empty list

Fidelius x 0

Fidelius out []

Illegibilus while $i < k$

WingardiumLeviosa **Entomorphis** i k **Imperio**

Alohomora

Illegibilus $x = x + \text{get}[i] \text{ fstStream}$

Fidelius x **Engorgio** x (**Accio** i **Legilimens** 0)

Illegibilus x : out

Depulso x out

Illegibilus i=i+1

Fidelius i **Engorgio** i 1

FiniteIncantatem

Illegibilus print out

Flagrate out

FiniteIncantatem

Problem 5 - Fibonacci

Take a sequence $a_1 a_2 a_3 a_4 a_5 \dots$ and output the sequence $a_1 a_1 + a_2 2a_1 + a_2 + a_3 3a_1 + 2a_2 + a_3 + a_4 5a_1 + 3a_2 + 2a_3 + a_4 \dots$ where the coefficients of each input term in the sums follows the Fibonacci series $1 1 2 3 5 8 \dots$ from when it first appears. Recall that the Fibonacci series starts with two 1s and then the subsequent terms are always the sum of the previous two.

pr5.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Fidelius n **Informous** (**Legilimens** 0)

Fidelius out []

Fidelius light **nox**

Confundo **Deprimo** n 2

Incendio **Alohomora**

Fidelius light **lumos**

Fidelius out **Depulso** (**Accio** 0 **Legilimens** 0) out

Fidelius out **Depulso** (**Ferula** (**Confringo** 0 1 **Legilimens** 0)) out

FiniteIncantatem

Aguamenti **Alohomora**

Confundo **Defodio** n o

Incendio **Alohomora**

Fidelius out **Depulso** (**Accio** 0 **Legilimens** 0) out

FiniteIncantatem

FiniteIncantatem

Fidelius i 2

Wingardium Leviosa (**Serpensortia** light (**Entomorphis** i n)) **Imperio**

Alohomora

Fidelius x 1

Fidelius y 1

Fidelius wand **Everte Statum** (**Confringo** 0 i **Legilimens** 0)

Fidelius s (**Ferula** (**Confringo** 0 1 wand))

Fidelius j 2

Wingardium Leviosa (**Entomorphis** j **Informous** wand) **Imperio**

Alohomora

Fidelius z **Engorgio** x y

Fidelius x y

Fidelius y z

Fidelius s **Engorgio** s **Geminio** y (**Accio** j wand)

Fidelius j **Engorgio** j 1

Finite Incantatem

Fidelius out **Depulso** s out

Fidelius i **Engorgio** i 1

Finite Incantatem

Flagrate out

Finite Incantatem

Problem 6 - Copying + Prefix

Take a sequence $a_1 a_2 a_3 a_4 a_5 \dots$ as an input and output two copies of it, the second prefixed with 0.

pr6.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Illegibilus the next statement stores in variable x the first stream from the file

Fidelius x **Legilimens** 0

Illegibilus the next two statements prints x

Flagrate x

Illegibilus the next statement adds 0 at the beginning of the (tail x) and prints the result

Flagrate Flipendo 0 **Ventus** x

Finite Incantatem

Problem 7 - Copying + Stream Arithmetic

Take two sequences $a_1 a_2 a_3 a_4 \dots$ and $b_1 b_2 b_3 b_4 \dots$, and produce two sequences
 $a_1 - b_1 a_2 - b_2 a_3 - b_3 a_4 - b_4 \dots$
 $a_1 a_2 a_3 a_4 \dots$

pr7.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Illegibilus the next statement checks if there are at least two input streams

Illegibilus parameters:

x - first stream

y - second stream

out - initialised empty output

i - index

MischiefManaged

Confundo Deprimo Pack 2 Incendio

Alohomora

Fidelius x Legilimens 0

Fidelius y Legilimens 1

Fidelius out []

Fidelius i 0

WingardiumLeviosa (Entomorphis i Informous x) Imperio

Alohomora

Illegibilus add the difference $x[i] - y[i]$ to the end of out

Depulso Reducio (Accio i x) (Accio i y) out

Fidelius i Engorgio i 1

FiniteIncantatem

Flagrate out

Flagrate x

FiniteIncantatem

Aguamenti Alohomora

Flagrate Legilimens 0

FiniteIncantatem

FiniteIncantatem

Problem 8 - Copying + Prefix + Stream Arithmetic

Take a sequence $a_1 a_2 a_3 a_4 a_5 \dots$ as an input and output $a_1 + 0 a_2 + a_1 a_3 + a_2 a_4 + a_3 a_5 + a_4 \dots$

pr8.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Illegibilus $x = 0$:(init fst_stream)

Fidelius x **Flipendo** 0 **Ventus Legilimens** 0

Fidelius i 0

Fidelius out []

WingardiumLeviosa Entomorphis i **Informous** x **Imperio**

Alohomora

Fidelius a (**Accio** i **Legilimens** 0)

Fidelius b (**Accio** i x)

Fidelius k **Engorgio** a b

Depulso k out

Fidelius i **Engorgio** i 1

FiniteIncantatem

Flagrate out

FiniteIncantatem

Problem 9 - Natural Numbers

Take a sequence $a_1 a_2 a_3 a_4 a_5 \dots$ as an input and output $a_1 2a_1 + a_2 3a_1 + 2a_2 + a_3 4a_1 + 3a_2 + 2a_3 + a_4 5a_1 + 4a_2 + 3a_3 + 2a_4 + a_5 \dots$

pr9.spl

We **Diana and Filip** solemnly swear that we are up to no good

Alohomora

Fidelius x **Legilimens** 0

Fidelius i 0

Fidelius k []

Fidelius out []

WingardiumLeviosa Entomorphis i Informous x Imperio

Alohomora

Depulso (Engorgio 1 i) k

Fidelius j 0

Fidelius n 0

WingardiumLeviosa CarpeRetractum j i Imperio

Alohomora

Fidelius n Engorgio n (Geminio (Accio j x) (Accio j EverteStatum k))

Fidelius j Engorgio j 1

FiniteIncantatem

Depulso n out

Fidelius i Engorgio i 1

FiniteIncantatem

Flagrate out

FiniteIncantatem

Problem 10 - Delayed Feedback

Take a sequence $a_1 a_2 a_3 a_4 a_5 a_6 \dots$ as an input and output $a_1 a_2 a_3 + a_1 a_4 + a_2 a_5 + a_3 a_6 + a_4 a_7 + a_5 a_8 + \dots$ that is, the first two outputs are the same as the first two inputs. Then, to produce the output at time $n > 2$, the current input a_n is added to the value which was previously output at time $n - 2$.

pr10.spl

We Diana and Filip solemnly swear that we are up to no good

Alohomora

Fidelius x Legilimens 0

Fidelius i 0

Fidelius out []

WingardiumLeviosa Entomorphis i Informous x Imperio

Alohomora

Fidelius n 0

Fidelius j i

WingardiumLeviosa Deprimo j 0 Imperio

Alohomora

Fidelius n Engorgio n (Accio j x)

Fidelius j Reducio j 2

FiniteIncantatem

Depulso n out
Fidelius i **Engorgio** i 1

FiniteIncantatem

Flagrate out
FiniteIncantatem