Pseudocode

HAI 1.2 CAN HAS STDIO?

I HAS A VAR ITZ "Hello, World!" VISIBLE VAR

KTHXBYE

COMMENTS	PSEUDOCODE
 Bound to syntax. Are part of the file's code. Used to clarify specific parts of the code 	 NOT bound to syntax. Can NOT be executed. Representation of the whole code's content Universal understanding for across programmers of different backgrounds.

What pseudocode looks like

```
procedure KEEP()
    read a string s from the input stream
    if s is not empty then
        output s // output what you just read two lines above
        KEEP( )
```

- Use the construct "**let** ... **be ...**" to initialize variables

let s be an empty string

...

let Q be an empty queue

enqueue x into Q

let S be an empty stack **push**

x onto S

let L be an empty list

append x to L

- Use the construct "**let** ... **be ...**" to initialize variables
- Use "=" to assign values to variables.

let s' be an empty string

let i = 1

let n = |s|

- Use the construct "**let** ... **be ...**" to initialize variables
- Use "=" to assign values to variables.
- To compare use =, \neq , <, >, \leq , \geq .
- To perform Arithmetic operations: +, -, ×, /, mod.

- Use the construct "**let** ... **be ...**" to initialize variables
- Use "=" to assign values to variables.
- To compare use =, \neq , <, >, \leq , \geq .
- To perform Arithmetic operations: +, -, ×, /, mod.
- Logical constants denotaion:

True	False	Negation	Conjudation	Disjunction	Non-existence
true	false	not	and	or	nil

- Use the construct "let ... be ..." to initialize variables
- Use "=" to assign values to variables.
- To compare use =, \neq , <, >, \leq , \geq .
- To perform Arithmetic operations: +, -, ×, /, mod.
- Logical constants denotaion:

True	False	Negation	Conjudation	Disjunction	Non-existence
true	false	not	and	or	nil

- Enumeration starts at 1 not 0.

- Use the construct "**let** ... **be** ..." to initialize variables
- Use "=" to assign values to variables.
- To compare use =, \neq , <, >, \leq , \geq .
- To perform Arithmetic operations: +, -, ×, /, mod.
- Logical constants denotaion:

True	False	Negation	Conjudation	Disjunction	Non-existence
true	false	not	and	or	nil

- Enumeration starts at 1 not 0.
- A Strings and characters are **NOT** introduced with quotes.

output Hello world!

- Use the construct "**let** ... **be** ..." to initialize variables
- Use "=" to assign values to variables.
- To compare use =, \neq , <, >, \leq , \geq .
- To perform Arithmetic operations: +, -, ×, /, mod.
- Logical constants denotaion:

True	False	Negation	Conjudation	Disjunction	Non-existence
true	false	not	and	or	nil

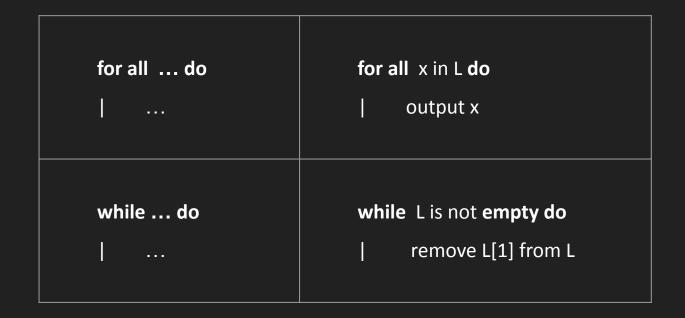
- A Enumeration starts at 1 not 0.
- Strings and characters are **NOT** introduced with quotes.
- Comments are introduced with "//"

Basic constructs

```
if ... then
else if ... then
else
```

```
function BIGGEST(a, b )
     if a > b then
           return a
     else if a < b then
           return b
     else
            return nil
```

Basic constructs



Basic constructs

repeat | ... until ...

repeat

output x - 1

until x = 0

Functions

To define a new function we need to use the following structure

function **NAME(** input_var)

- 1. Array
- 2. List
- 3. Stack
- 4. Queue
- 5. Priority Queue
- 6. Set
- 7. Matrix
- 8. Dictionary

Github repository Pseudocode/**README.md**

6. Set

A set is a collection of **unique elements** <u>without an order</u>. When defining a set you can use the mathematical notation:

- To denote x **belonging** to a set S use $x \in S$

6. Set

A set is a collection of **unique elements** <u>without an order</u>. When defining a set you can use the mathematical notation:

- To denote x **belonging** to a set S use $x \in S$

The **member** function returns **true** if the element

belongs to the set and false otherwise

let S be an empty set output member(x, S) //output is "false"

6. Set

A set is a collection of **unique elements** <u>without an order</u>. When defining a set you can use the mathematical notation:

- To denote x **belonging** to a set S use $x \in S$
- To insert an element x onto a set S use S = S U {x}
- To delete an element x from a set S use S = S \ {x}

6. Set

A set is a collection of **unique elements** <u>without an order</u>. When defining a set you can use the mathematical notation:

- To denote x **belonging** to a set S use $x \in S$
- To insert an element x onto a set S use S = S U {x}
- To delete an element x from a set S use $S = S \setminus \{x\}$.

let S be an empty set

→ insert x into S

for all $x \in S$ do

delete x from S

Build a function called *num_vocals* that given a string (s) in lower case, it returns the number of vocals in s.

Build a function called *num_vocals* that given a string (s) in lower case, it returns the number of vocals in s.

function **NUM-VOCALS**(s) return . . .

Build a function called *num_vocals* that given a string (s) in lower case, it returns the number of vocals in s.

function **NUM-VOCALS**(s)

 $\mathbf{nv} = 0$

• • •

return nv

Build a function called *num_vocals* that given a string (s) in lower case, it returns the number of vocals in s.



Build a function called *num_vocals* that given a string (s) in lower case, it returns the number of vocals in s.

```
function NUM-VOCALS(s)
     \mathbf{nv} = 0
     for all i in |s| do
           if s[i] \in \{a, e, i, o, u\} then
     return nv
```

"∈" denotes
belonging

Build a function called *num_vocals* that given a string (s) in lower case, it returns the number of vocals in s.

```
function NUM-VOCALS(s)
     \mathbf{nv} = 0
     for all i in |s| do
           if s[i] \in \{a, e, i, o, u\} then
                 nv = nv + 1
     return nv
```

Activity 1. Block 1 problems

Input: GATTACA
Output: GAUUACA

Problem 2. Find coding region:

Input: ATGGATTACATGATTT
Output: GATTACA

Problem 3. GC content:

Input: **GCAAATTTTT**Output: **20%**

Problem 4. Flip a string:

Input: STRESSED
Output: DESSERTS

- 1. Array
- 2. List
- 3. Stack
- 4. Queue
- 5. Priority Queue
- 6. Set
- 7. Matrix
- 8. Dictionary

2. List

A list is a sequence of elements arranged in a single dimension. It has **no** predetermined size and only the first and last elements can be accessed.

functions	definition	example
front	returns the first element in the list	output front L
back	returns the last element in the list	output back L
prev	returns the element before a given element	L.prev(i)
next	returns the element after a given element	L.next(i)
append	inserts an element at the back of the list	append x to L
concatenate	operation deletes the elements of another list and inserts them at the end of the first	concatenate L' to L

2. List

A list is a sequence of elements arranged in a single dimension. It has **no** predetermined size and only the first and last elements can be accessed.

```
let L be a list 1 to 3

let L' be an empty list

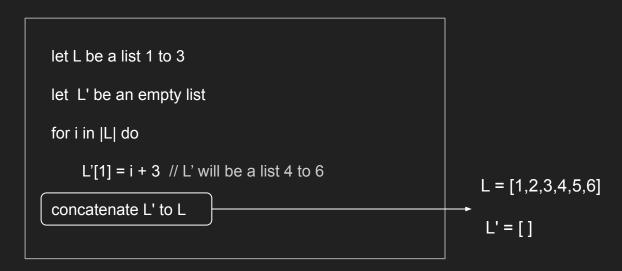
for i in |L| do

L'[1] = i + 3 // L' will be a list 4 to 6

concatenate L' to L
```

2. List

A list is a sequence of elements arranged in a single dimension. It has **no** predetermined size and only the first and last elements can be accessed.



8. Dictionary

A dictionary is a structure containing a *set* of **keys** each of which has an associated **value**. The **value** of a given **key** is accessed using D[key].

functions	definition	example
member	returns true if the key is in the dictionary, false otherwise.	member(x, D)
lookup	returns the element associated with the given key in the dictionary, nil if not found.	D[x] lookup
insert	inserts an element with a given key and information into the dictionary, replacing the element (if any) with the given key.	D[x] = y
delete	deletes the the given key from the dictionary along with its value.	delete x from D

8. Dictionary

A dictionary is a structure containing a *set* of **keys** each of which has an associated **value**. The **value** of a given **key** is accessed using D[key].

```
let D be a dictionary

for all x \in D do

y = D[x] lookup

output (x, y)

delete x from D //removes key x and its value
```

Activity 2. Block 2 problems

Problem 5. Find maximum		Problem 6. Pizza count			
		Input:	Output:		
Input: 1 2 4 5 8	Output: 8	Pepperoni Hawaiian BBQ Veggie Hawaiian BBQ Cheeses BBQ	Cheeses: 1 Hawaiian: 2 Pepperoni: 1 Veggie: 1 BQQ: 2		
Problem 7. ESCI co	Problem 7. ESCI colors		Problem 8. Fill the rows		
Input:	Output:	Input:	Output:		
Blue Red Green Orange Violet Violet Green	Green Orange Red Violet	Anna A. Jordi A. Jorge B. Marc C. Antonio F. Violet P. Adria P. Maria Z.	Row 1: Anna A., Marc C., Adria P. Row 2: Jordi A., Antonio F., Maria Z. Row 3: Jorge B., Violet P.		