layout plan Coe428

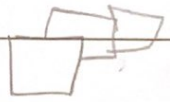
1 → must be < > ! → Done → ch

2. Detect when tag is inputted

1 <a> x → test 1

2. <a> <c> <d> </d> </c> ✓

TESTING



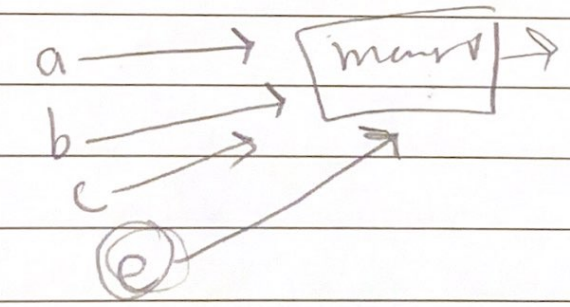
pop

<a> <c>

stack [a, b, c, D, e] pop

PFC popped: e → removes e

[a b c D]



Ahraz

Ahraz

IF [a, b, c, D, E]

Previous has to be same

[a b a]

[a b b a]

<a><c><d><e> →

b
a
c
b
a

0 → 3

a

- IF (Something to push) not in array its self → or array 0

[E b c D E]a]E

0 1 2 3

- the value itself prior or value of First array Variable.
- element must be in array already

hypotosis:

1. Remove the Previous array Variable
2. Remove First → Wrong

Test <a>

↳ Proper test but is incorrect, Can not move a before b in this case

↳ solution:

↳ only be able to close the Previous array elent

<a><c><d> ↓ remove

<a><a>

a[b[a-a[b-b]]-b]-a

<a><a> → gone

<a> → Same

<a> → Same

<a> → Same

∴ only rule 1. holds and rule 2 fails

∴ if (Something2Push = stack top)

↳ try

Coelab5 →

Conclusion:

I. only previous element can be compared and escaped

→ last entry

IF(strcmp(stack[top-1], Value) == 0)

Test 1: <a><b><c><d><d><c><b><a>

<a><b><c><c><b><a>

<a><b><b><a>

<a><a> → empty → ✓

Issue #1 → 1st error → Segmentation error by index out of bounds

↳ Fix → add IF statement which doesn't send pop() back IF i<0

↳ Fixed

Issue #2 → implementation of is empty

Idea: add IF statement to pop IF empty

↳ Fixed ✓

Requirement 2 → part 2 → analysis

Part 2:

- push() is related to addheap()
- pop() is related to deleteheap()
- isEmpty() → own thing
- heapsize() → own thing

Steps:

Step 1: send values to push() and test array

Step 2: push() send values to addheap() to create

Step 3: Fix pop() which will affect deleteheap()

Correction, I may have been extra by accident by part2main → intstack → intheap

step 1 ☑ Step 2 ☑

Steps Revision:

Step 3: heap sort the values from largest to smallest

addheap() → builds the max heap → R sort

delete heap sorts it

Next Step → learn max-heapify and input

addheap = maxheapify

part2main → push() → addheap()

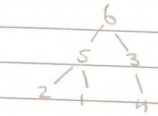
pop() ← deleteheap

Cse 4288 lab 5

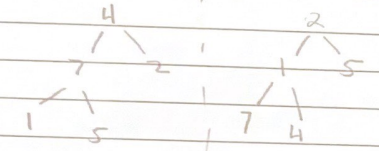
part 3:

get number [6 5 3 2 1 4]

Testing to find error



debugging



5 // 2.5 → 2

[4 7 2 1 5]

4 → 5

true

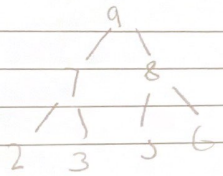
[4 7 2 1 5]

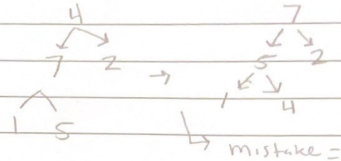
1 > 7 → false

max = left

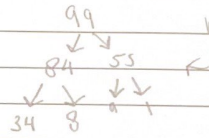
40 > 65 but 80

39 > 32 but 47 → 6





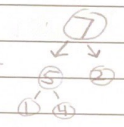
mistake =



mistake → switch needed

og: [4 7 2 1 5]

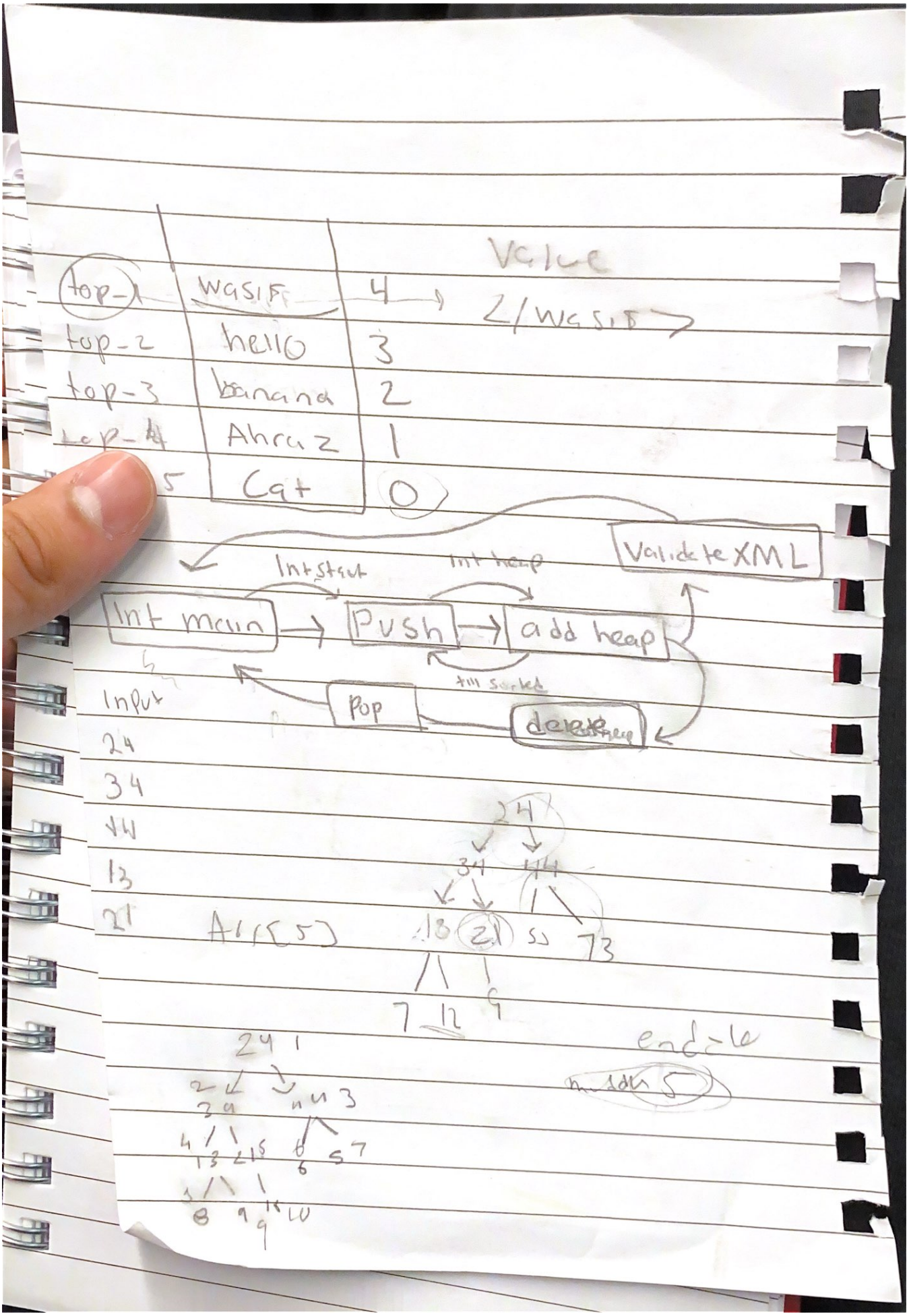
N: [7 5 2 1 4]



part 2 main → int stack → push → add stacks

Rep ← done

<Parent tag>


Value
top-1
wasIF
4
</wasIF>
top-2
hello
3
top-3
banana
2
top-4
Ahruz
1
5
Cat
0
Validate XML
Int struct
Int heap
Int main
Push
add heap
full sorted
Pop
delete
Input
24
34
44
13
21
Arr[5]
end=6

Requirement 2.

Values Values
PartAMain → Int Stack → add heap

Numbers get sent to int Stack
↳ int Stack does some Funtions and
Stacks the Variables.
then Variables most likely in an array are
sent to → int heap where
it sorted and reverse sorted

Code 1: runs → numbers collected
- 1, 5, 3, 7, 9, 2 → Randomly, not array sorted
Code 2: collects numbers, and inserts into array
using Push() PS → pop Funtion unknown
Code 3: Sorts in max heap and heap sort
addheap() → Sorts in max heap by
deleteheap() → Sorts in heap sort
heapsize() → returns total number of events