Stacks and Subroutines

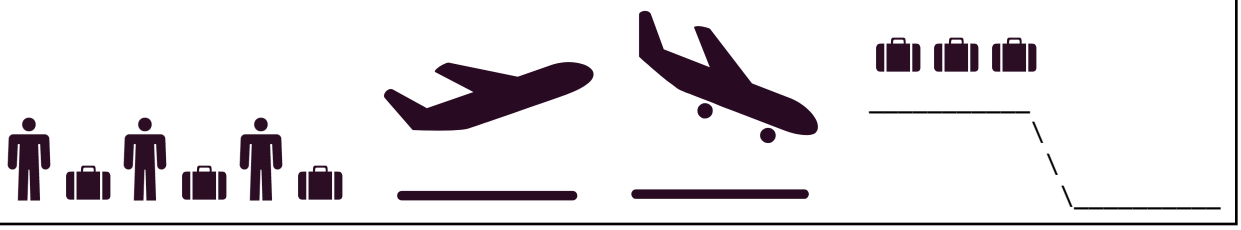
Stack

_____ data structure - _____ _____ _____ _____

_____ - _____

_____ - _____

Airplane Luggage Example



Matching Parentheses

```
Given a string containing just the characters '(', ')', '{', '}', '[' and ']',
determine if the string contains all valid sets of parentheses.

Sample inputs / outputs:









Algorithm
Loop through elements in string:

    If _____________________________________________________________:
        Push onto stack

    If _____________________________________________________________:
        Pop last item from stack

            If _____________________________________________________:
                Return False

Return True
```

Input: “()”

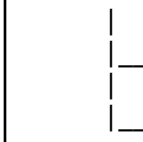
Character : “(”

Character : “)”

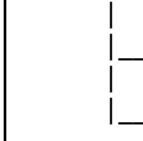
Input: “{[]}”

Character : “{”

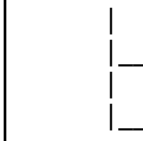
Character : “[”



Character : “]”

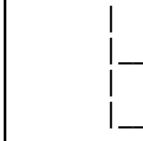


Character : “}”



Input: “[)”

Character : “[”



Character : “)”

Input: “([)]” Character : “[”	
Character : “)”	
Input: “([)]” Character : “(”	
Character : “[”	
Character : “)”	
Character : “]”	

Subroutine

aka: _____

We can use the stack to _____

so we can retrieve them _____

In MIPS, the stack grows _____

Register Review

$t0 - $t9
$s0 - $s7
$a0 - $a3
$v0 - $v1
$ra
$sp

Subroutine Diagram

Stack Usage

Push One Register

How to push a register value to the stack:

Example

```
Push $s0 to stack
ex/ $s0 = 0xAAAAAAAA
```

$sp ➔

ADDRESS	CONTENTS
0x7FFF FE00	0xDEADBEEF
0x7FFF FDFC	
0x7FFF FDF8	
0x7FFF FDF4	

Pop One Register

How to pop a register value from the stack:

Example

```
Pop one word from stack
```

$sp ➔

ADDRESS	CONTENTS
0x7FFF FE00	0xDEADBEEF
0x7FFF FDFC	
0x7FFF FDF8	
0x7FFF FDF4	

Push Multiple Registers

How to push multiple registers to the stack:

Example

Push $s0, $s1, $s2 to stack

ex/ $s0 = 0xAAAAAAAA
$s1 = 0xBBBBBBBB
$s2 = 0xCCCCCCCC

$sp
➔

ADDRESS	CONTENTS
0x7FFF FE00	0xC0FFEEEE
0x7FFF FDFC	
0x7FFF FDF8	
0x7FFF FDF4	

One at a time:

Decrement $sp once:

Pop Multiple Registers

How to pop multiple registers from the stack:

Example

Pop 3 words from stack

$sp ➔

ADDRESS	CONTENTS
0x7FFF FE00	0xC0FFEEEE
0x7FFF FDFC	0xAAAAAAAA
0x7FFF FDF8	0xBBBBBBBB
0x7FFF FDF4	0xCCCCCCCC

One at a time:

Increment $sp once:

Jump Instructions

J: _____

Give an example of how to use this instruction

J _____

What does this instruction do?

How does this instruction affect the registers?

JAL: _____

Give an example of how to use this instruction

JAL _____

What does this instruction do?

How does this instruction affect the registers?

JALR: _____

Give an example of how to use this instruction

JALR _____

What does this instruction do?
How does this instruction affect the registers?

JR: _____

Give an example of how to use this instruction JR _____
What does this instruction do?
How does this instruction affect the registers?

Examples

Basic Example

address of instruction	instruction
0x100	Main: NOP
0x104	JAL Sub1
0x108	NOP
0x10C	JAL Sub2
0x110	NOP
...	...
0x200	Sub1: NOP
0x204	JR $ra
...	...
0x300	Sub2: NOP
0x304	JR $ra

time	current pc	ra (after instr)	pc (after instr)
1			
2			
3			
4			
5			
6			
7			
8			
9			

Nested Subroutine

address of instruction	instruction
0x100	Main: NOP
0x104	JAL Sub2
0x108	NOP
...	...
0x200	Sub1: NOP
0x204	JR $ra
...	...
0x300	Sub2: NOP
0x304	JAL Sub1
0x308	NOP
0x30C	NOP
0x310	JR $ra

time	current pc	ra (after instr)	pc (after instr)
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			

Which subroutines are the callers?

Which are the callees?

Is there a problem with this code?

Callee Save Registers

Caller Save Registers

Callee Save Example

Use the stack to save $ra

address of instruction	instruction
0x100	Main: NOP
0x104	JAL Sub2
0x108	NOP
...	...
0x200	Sub1: NOP
0x204	JR $ra
...	...
0x300	Sub2: NOP
0x304	
0x308	
0x30C	JAL Sub1
0x310	NOP
0x314	NOP
0x318	
0x31C	
0x310	JR $ra
...	...

time	pc (crnt instr)	ra (after instr)	pc (after instr)	memory address (in stack)	contents
1				0xFE00	
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					