Vocab

Three MIPS steps: Fetch, Decode, Execute

C Compiler is preprocessor and translator

Linker assembles all functions and make final exe

Loader loads program into memory

Constant Type

29LU = unsigned long integer

"t" = string

-39.54Lf = long double

7hd = short integer

Memory Allocation

```
Int A = 200;
float B[2][2] = {{1.0,2.0}, {3.0,4.0}};
float *p2 = &(B[1][1]);
int *p1 = &A;
char s[ ] = "foo";
```

2000	A				
2004	B[0][0]				
2008	B[0][1]				
2012	B[1][0]				
2016	B[1][1]				
2020					
2024	p2				
2028					
2032	p1				
2036					
2040	S[0]	S[1]	S[2]	S[3]	

Values

(p1 = 2000)(*p1 = 200)(&p1 = 2032)(p2 = 2016)

''*p2 = 4)(&p2 = 2024)(&(s[1]) = 2041)*((s+3) = 0)

Loop Array C

int A[]; int i = 0; do {p = p*A[i]; i++;} while (A[i]);

Swap

1. (7 pts) Write a MIPS code fragment that adds the value 0x4FA2C3 to register $1 and puts the result in $2. Modify only registers $1 and $2. (You may use hexadecimal immediate values.) *For maximum credit, include comments.*

Label	Instruction	Comment
	`lui $2, 0x4F`	`# load upper 16 bits`
	`ori $2, $2, 0xA2C3`	`# load rest of constant`
	`add $2, $2, $1`	`# add it to $1`

Swap MIPS

```
void swap(int v[], int k)
{
    int temp;
    temp = v[k]
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

Parameters:
$2 = Address of v[ ]
$3 = k, and
Return address is in **$31**

Label	Instruction	Comment
Swap:	sll $4, $3, 2	# $4 = k*4
	add $4, $4, $2	# $4=v+k*4
	lw $5, 0($4)	# $5=v[k]
	lw $6, 4($4)	# $6=v[k+1]
	sw $6, 0($4)	# v[k]=$6
	sw $5, 4($4)	# v[k+1]=$5
	jr $31	# return

Values after

/ x = 7, y = 7, and z = 2/

if (z = x < y) { x += 3; y -= 1; }

else x = y++;

x = 7, y = 8, z = 0

Loop C to Mips

```
Sum = 0;
while (N!=0){
    Sum += N&1;
    N = N >> 1;
}
```

Label	Instruction	C-comment
	add $2, $0, 0	# Sum = 0
Loop:	beq $1, $0, Exit	# exit if N == 0
	and $3, $1, 1	# $3: N&1
	add $2, $2, $3	# Sum +=N&1
	srl $1, $1, 1	# N = N >>1
	j Loop	# continue looping
Exit:	...	