

## CS-410 Software Reverse Engineering – Assignment 6

Binary to C++ With Security Vulnerabilities

Eric Slutz

Southern New Hampshire University



<b>Blocks of Assembly Code</b>	Explanation of Functionality
push %rbp	Start program and begin looping until %eax comparison to 5 is true.
mov %rsp,%rbp	
sub \$0x20,%rsp	
mov %fs:0x28,%rax	
mov %rax,-0x8(%rbp)	
xor %eax,%eax	
movl \$0x0,-0x14(%rbp)	
mov -0x14(%rbp),%eax	
cmp \$0x5,%eax	
je 0x308 <main+655></main+655>	
lea 0x0(%rip),%rsi	Displays menu
lea 0x0(%rip),%rdi	
callq 0xb6 <main+61></main+61>	
lea 0x0(%rip),%rsi	
lea 0x0(%rip),%rdi	
callq 0xc9 <main+80></main+80>	
lea 0x0(%rip),%rsi	
lea 0x0(%rip),%rdi	
callq 0xdc <main+99></main+99>	
lea 0x0(%rip),%rsi	
lea 0x0(%rip),%rdi	
callq 0xef <main+118></main+118>	
lea 0x0(%rip),%rsi	Gets menu selection
lea 0x0(%rip),%rdi	
callq 0x102 <main+137></main+137>	
lea 0x0(%rip),%rsi	Gets first user input
lea 0x0(%rip),%rdi	
callq 0x115 <main+156></main+156>	
lea -0x14(%rbp),%rax	Gets second user input
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x128 <main+175></main+175>	
mov -0x14(%rbp),%eax	Performs subtraction on user input and outputs the result in a
cmp \$0x1,%eax	formatted statement.
jne 0x1c9 <main+336></main+336>	
lea -0x10(%rbp),%rax	
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x147 <main+206></main+206>	
mov %rax,%rdx	
lea -0xc(%rbp),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x159 <main+224></main+224>	
mov -0x10(%rbp),%eax	
mov %eax,%esi	
lea 0x0(%rip),%rdi	



calla Or16a cmain 241	
callq 0x16a <main+241></main+241>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x179 <main+256></main+256>	
mov %rax,%rdx	
mov -0xc(%rbp),%eax	
mov %eax,%esi	
mov %rdx,%rdi	
callq 0x189 <main+272></main+272>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x198 <main+287></main+287>	
mov %rax,%rcx	
mov -0x10(%rbp),%edx	
mov -0xc(%rbp),%eax	
sub %eax,%edx	
mov %edx,%eax	
mov %eax,%esi	
mov %rcx,%rdi	
callq 0x1af <main+310></main+310>	
mov %rax,%rdx	
mov 0x0(%rip),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x1c4 <main+331></main+331>	
jmpq 0x97 <main+30></main+30>	
mov -0x14(%rbp),%eax	Performs addition on user input and outputs the result in a formatted
cmp \$0x2,%eax	statement.
jne 0x268 <main+495></main+495>	
lea -0x10(%rbp),%rax	
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x1e8 <main+367></main+367>	
mov %rax,%rdx	
lea -0xc(%rbp),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x1fa <main+385></main+385>	
mov -0x10(%rbp),%eax	
mov %eax,%esi	
lea 0x0(%rip),%rdi	
lea 0x0(%rip),%rdi callq 0x20b <main+402></main+402>	
lea 0x0(%rip),%rdi callq 0x20b <main+402></main+402>	
lea 0x0(%rip),%rdi callq 0x20b <main+402> lea 0x0(%rip),%rsi mov %rax,%rdi</main+402>	
lea 0x0(%rip),%rdi callq 0x20b <main+402> lea 0x0(%rip),%rsi mov %rax,%rdi</main+402>	
lea 0x0(%rip),%rdi callq 0x20b <main+402> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x21a <main+417> mov %rax,%rdx</main+417></main+402>	
lea 0x0(%rip),%rdi callq 0x20b <main+402> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x21a <main+417> mov %rax,%rdx</main+417></main+402>	



mov %rdx,%rdi	
callq 0x22a <main+433></main+433>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x239 <main+448></main+448>	
mov %rax,%rcx	
mov -0x10(%rbp),%edx	
mov -0xc(%rbp),%eax	
add %edx,%eax	
mov %eax,%esi	
mov %rcx,%rdi	
callq 0x24e <main+469></main+469>	
mov %rax,%rdx	
mov 0x0(%rip),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x263 <main+490></main+490>	
jmpq 0x97 <main+30></main+30>	
mov -0x14(%rbp),%eax	Performs division on user input and outputs the result in a formatted
cmp \$0x3,%eax	statement.
jne 0x97 <main+30></main+30>	
lea -0x10(%rbp),%rax	
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x287 <main+526></main+526>	
mov %rax,%rdx	
lea -0xc(%rbp),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x299 <main+544></main+544>	
mov -0x10(%rbp),%eax	
mov %eax,%esi	
lea 0x0(%rip),%rdi	
callq 0x2aa <main+561></main+561>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x2b9 <main+576></main+576>	
mov %rax,%rdx	
mov -0xc(%rbp),%eax	
mov %eax,%esi	
mov %rdx,%rdi	
callq 0x2c9 <main+592></main+592>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x2d8 <main+607></main+607>	
mov %rax,%rcx	
mov -0x10(%rbp),%eax	
mov -0xc(%rbp),%esi	
шо. оло(/отор),/осы	



cltd	
idiv %esi	
mov %eax,%esi	
mov %rcx,%rdi	
callq 0x2ee <main+629></main+629>	
mov %rax,%rdx	
mov 0x0(%rip),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x303 <main+650></main+650>	
jmpq 0x97 <main+30></main+30>	
mov \$0x0,%eax	Resets value
mov -0x8(%rbp),%rcx	Loops back to beginning
xor %fs:0x28,%rcx	
je 0x321 <main+680></main+680>	
callq 0x321 <main+680></main+680>	Exits program
leaveq	
retq	



Blocks of Assembly Code	C++ Code
•	int menuSelection = 0, num1, num2;
cmp \$0x5,%eax	while (menuSelection < 5)) {
je 0x308 <main+655></main+655>	
lea 0x0(%rip),%rsi	DisplayMenu();
lea 0x0(%rip),%rdi	
callq 0xb6 <main+61></main+61>	
lea 0x0(%rip),%rsi	
lea 0x0(%rip),%rdi	
callq 0xc9 <main+80></main+80>	
lea 0x0(%rip),%rsi	
lea 0x0(%rip),%rdi	
callq 0xdc <main+99></main+99>	
lea 0x0(%rip),%rsi	
lea 0x0(%rip),%rdi	
callq 0xef <main+118></main+118>	
lea 0x0(%rip),%rsi	std::cin >> menuSelection;
lea 0x0(%rip),%rdi	
callq 0x102 <main+137></main+137>	
lea 0x0(%rip),%rsi	std::cin >> num1;
lea 0x0(%rip),%rdi	
callq 0x115 <main+156></main+156>	
lea -0x14(%rbp),%rax	std::cin >> num2;
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x128 <main+175></main+175>	
mov -0x14(%rbp),%eax	if (menuSelection == 1) {
cmp \$0x1,%eax	std::cout << num1 << " + " << num2 << " = " <<
jne 0x1c9 <main+336></main+336>	num1 + num2 << std::endl;



lea	1 0 10 (0 1 ) 0	,
lea 0x0(%rip),%rdi callq 0x147 <main+206> mov %rax,%rdx lea -0xc(%rbp),%rax mov %rax,%rsi mov %rdx,%rdi callq 0x159 <main+224> mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rax,%rdx mov -0xc(%rbp),%eax mov %rax,%rdx mov -0xc(%rbp),%eax mov %rax,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+287> mov %rax,%rcx mov -0xc(%rbp),%eax mov -0xc(%rbp),%eax</main+287></main+272></main+256></main+241></main+224></main+206>		}
callq 0x147 <main+206> mov %rax,%rdx lea -0xc(%rbp),%rax mov %rax,%rsi mov %rdx,%rsi mov wrdx,%rsi mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov weax,%esi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%eax mov -0x10(%rbp),%eax</main+287></main+272></main+272></main+272></main+256></main+241></main+206>	•	
mov %rax,%rdx lea -0xc(%rbp),%rax mov %rax,%rsi mov %rdx,%rdi callq 0x159 <main+224> mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+287> mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0xc(%rbp),%eax</main+287></main+287></main+272></main+272></main+256></main+241></main+224>		
lea -0xc(%rbp),%rax mov %rax,%rsi mov %rdx,%rdi callq 0x159 <main+224> mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+287> mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%edx mov -0xc(%rbp),%edx</main+287></main+287></main+272></main+272></main+256></main+241></main+224>	•	
mov %rax,%rsi mov %rdx,%rdi callq 0x159 <main+224> mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%edx mov -0xc(%rbp),%edx</main+287></main+272></main+272></main+256></main+241></main+224>		
mov %rdx,%rdi callq 0x159 <main+224> mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%edx mov -0xc(%rbp),%edx</main+287></main+272></main+272></main+256></main+241></main+224>	* * *	
callq 0x159 <main+224> mov  -0x10(%rbp),%eax mov  %eax,%esi lea  0x0(%rip),%rdi callq 0x16a <main+241> lea  0x0(%rip),%rsi mov  %rax,%rdi callq 0x179 <main+256> mov  %rax,%rdx mov  -0xc(%rbp),%eax mov  %eax,%esi mov  %rdx,%rdi callq 0x189 <main+272> lea  0x0(%rip),%rsi mov  %rax,%rdi callq 0x189 <main+272> lea  0x0(%rip),%rsi mov  %rax,%rdi callq 0x198 <main+287> mov  %rax,%rcx mov  -0x10(%rbp),%edx mov  -0xc(%rbp),%eax</main+287></main+272></main+272></main+256></main+241></main+224>		
mov -0x10(%rbp),%eax mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272></main+256></main+241>	· · · · · · · · · · · · · · · · · · ·	
mov %eax,%esi lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272></main+256></main+241>	•	
lea 0x0(%rip),%rdi callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%edx mov -0xc(%rbp),%edx</main+287></main+272></main+256></main+241>	* **	
callq 0x16a <main+241> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272></main+256></main+241>		
lea  0x0(%rip),%rsi mov  %rax,%rdi callq  0x179 <main+256> mov  %rax,%rdx mov  -0xc(%rbp),%eax mov  %eax,%esi mov  %rdx,%rdi callq  0x189 <main+272> lea  0x0(%rip),%rsi mov  %rax,%rdi callq  0x198 <main+287> mov  %rax,%rcx mov  -0x10(%rbp),%edx mov  -0xc(%rbp),%eax</main+287></main+272></main+256>	_	
mov %rax,%rdi callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272></main+256>	-	
callq 0x179 <main+256> mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272></main+256>	* * * * * * * * * * * * * * * * * * * *	
mov %rax,%rdx mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272>		
mov -0xc(%rbp),%eax mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272>	•	
mov %eax,%esi mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272>	•	
mov %rdx,%rdi callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272>	171	
callq 0x189 <main+272> lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287></main+272>		
lea 0x0(%rip),%rsi mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287>	•	
mov %rax,%rdi callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287>	•	
callq 0x198 <main+287> mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax</main+287>	lea 0x0(%rip),%rsi	
mov %rax,%rcx mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax	·	
mov -0x10(%rbp),%edx mov -0xc(%rbp),%eax		
mov -0xc(%rbp),%eax		
	mov -0x10(%rbp),%edx	
sub %eax.%edx	* * *	
	sub %eax,%edx	
mov %edx,%eax	mov %edx,%eax	
mov %eax,%esi	mov %eax,%esi	
mov %rcx,%rdi	mov %rcx,%rdi	
callq 0x1af <main+310></main+310>	callq 0x1af <main+310></main+310>	
mov %rax,%rdx	mov %rax,%rdx	
mov 0x0(%rip),%rax	mov 0x0(%rip),%rax	
mov %rax,%rsi	mov %rax,%rsi	
mov %rdx,%rdi	mov %rdx,%rdi	
callq 0x1c4 <main+331></main+331>	callq 0x1c4 <main+331></main+331>	
jmpq 0x97 <main+30></main+30>	jmpq 0x97 <main+30></main+30>	
mov -0x14(%rbp),%eax else if (menuSelection == 2) {		else if (menuSelection == 2) {
cmp \$0x2,%eax std::cout << num1 << " - " << num2 << " = " <<	cmp \$0x2,%eax	std::cout << num1 << " - " << num2 << " = " <<
jne 0x268 <main+495> num1 - num2 &lt;&lt; std::endl;</main+495>	jne 0x268 <main+495></main+495>	



1 0 10/0/1 > 0/	,
lea -0x10(%rbp),%rax	}
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x1e8 <main+367></main+367>	
mov %rax,%rdx	
lea -0xc(%rbp),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x1fa <main+385></main+385>	
mov -0x10(%rbp),%eax	
mov %eax,%esi	
lea 0x0(%rip),%rdi	
callq 0x20b <main+402></main+402>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x21a <main+417></main+417>	
mov %rax,%rdx	
mov -0xc(%rbp),%eax	
mov %eax,%esi	
mov %rdx,%rdi	
callq 0x22a <main+433></main+433>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x239 <main+448></main+448>	
mov %rax,%rcx	
mov -0x10(%rbp),%edx	
mov -0xc(%rbp),%eax	
add %edx,%eax	
mov %eax,%esi	
mov %rcx,%rdi	
callq 0x24e <main+469></main+469>	
mov %rax,%rdx	
mov 0x0(%rip),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x263 <main+490></main+490>	
jmpq 0x97 <main+30></main+30>	
mov -0x14(%rbp),%eax	else if (menuSelection == 3) {
cmp \$0x3,%eax	std::cout << num1 << " * " << num2 << " = " <<
ine 0x97 <main+30></main+30>	num1 * num2 << std::endl;
J	ALVANIA - NO DOMINATIONS



100 Or 10(0/ mbm) 0/ morr	1
lea -0x10(%rbp),%rax	}
mov %rax,%rsi	
lea 0x0(%rip),%rdi	
callq 0x287 <main+526></main+526>	
mov %rax,%rdx	
lea -0xc(%rbp),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x299 <main+544></main+544>	
mov -0x10(%rbp),%eax	
mov %eax,%esi	
lea 0x0(%rip),%rdi	
callq 0x2aa <main+561></main+561>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x2b9 <main+576></main+576>	
mov %rax,%rdx	
mov -0xc(%rbp),%eax	
mov %eax,%esi	
mov %rdx,%rdi	
callq 0x2c9 <main+592></main+592>	
lea 0x0(%rip),%rsi	
mov %rax,%rdi	
callq 0x2d8 <main+607></main+607>	
mov %rax,%rcx	
mov -0x10(%rbp),%eax	
mov -0xc(%rbp),%esi	
cltd	
idiv %esi	
mov %eax,%esi	
mov %rcx,%rdi	
callq 0x2ee <main+629></main+629>	
mov %rax,%rdx	
mov 0x0(%rip),%rax	
mov %rax,%rsi	
mov %rdx,%rdi	
callq 0x303 <main+650></main+650>	
jmpq 0x97 <main+30></main+30>	
	}