Q1:

```
min:
        push ebp                    //Save old frame pointer
        mov ebp, esp                //Initialize new frame
        sub esp, 4                  //Allocate space for local variables
        push ebx                    //Save non-volatile registers
        mov eax, [ebp+8]            //eax = a
        mov [ebp-4], eax            //V = a
        mov eax, [ebp+12]           //eax = b
        cmp eax, [ebp-4]            //b - V
        jge min0                    //jump if b >= V
        mov [ebp-4], eax            //V = b
min0:
        mov eax, [ebp+16]           //eax = c
        cmp eax, [ebp-4]            //c - V
        jge min1                    //jump if c >= V
        mov [ebp-4], eax
min1:
        mov eax, [ebp-4]            //return V
        pop ebx                     //restore non-volatile registers
        mov esp, ebp                //restore stack pointer
        pop ebp                     //restore frame pointer
        ret 0
```

```
p:
        push ebp
        mov ebp, esp
        sub esp, 4
        push ebx
        push [ebp+12]               //Pass parameters right to left
        push [ebp+8]                //to min function
        push [g]
        call min
        add esp, 12                 //Remove parameters from stack
        mov [ebp-4], eax            //V = min(...)
        push [ebp+20]               //Again pass parameters R->L to
        push [ebp+16]               //min function
        push [ebp-4]
```

```
call   min
add    esp, 12     // Remove parameters from stack
pop    ebx         // Restore non-volatile registers
mov    esp, ebp    // Restore stack pointer
pop    ebp         // Restore frame pointer
ret 0              // Note return value from min is
                   // already in eax
```

```
gcd:
    push   ebp
    mov    ebp, esp
    push   ebx
    mov    eax, [ebp+12]
    test   eax, eax          // Test for eax == 0
    jne gcd0
    mov eax, [ebp+8]
    pop    ebx
    mov    esp, ebp
    pop ebp
    ret 0
gcd0:
    mov    eax, [ebp+8]
    and    edx, 0      // Clear edx
    mov ebx, [ebp+12]  // ebx = b
    div ebx    // eax = eax / ebx with edx = eax % ebx
    push   edx  // Push parameters onto stack (b)
    push   ebx  // (a)
    call   gcd
    add    esp, 8
    pop    ebx
    mov    esp, ebp
    pop    ebp
    ret 0
```

Q2:

Stack Frames

gcd(14, 21) 1

gcd(21, 14) 2

gcd(14, 7) 3

gcd(7, 0) 4

b = 0 return

p(8, 9, 3, 2)

min(4, 8, 9)

↳ 4

min(4, 3, 2)

↳ 2