## Question 1

1 / 1 pts

Complete the symbol table entries for the parameters and variables in the following **Jack** function:

```
function int blowupstack(int c, int z)  arg
{
        var string message ;  } local
        var int d, a ;

        let c = blowupstack(a,z) ;
        return z ;
}
```

The symbol has four columns:

Identifier, type, segment, offset.

**Correct!**

**a, int,**               local,2

**Correct!**

**c, int,**               argument, 0

**Correct!**

**d, int,**               local,1

**Correct!**

**z, int,**               argument,1

**Correct!**

**message, string,**      local, 0

Other Incorrect Match Options:
- argument, 3
- this, 3
- local, 4
- this, 2
- this, 1
- this, 4
- this, 0
- local, 3
- argument, 2
- argument, 4

## Question 2

Complete the symbol table entries for the parameters and variables in the following **Jack** methods:

```
class dummy
{
    function int sillyb(int a, int z)    arg
    {
        var int f, c ;    local
        return f + c ;
    }
    method int sillya(int g, int b)    arg
    {
        var int d, e ;    local
        return d + e * b ;
    }
}
```

The symbol has four columns:

Identifier, type, segment, offset.

| a | arg | 0 |
|---|-----|---|
| z | arg | 1 |
| f | local | 0 |
| c | local | 1 |

| g | arg | 1 |
|---|-----|---|
| b | arg | 2 |
| d | local | 0 |
| e | local | 1 |

this dummy argument 0

| Correct! | a, int, | argument, 0 |
|----------|---------|-------------|
| Correct! | b, int, | argument, 2 |
| Correct! | c, int, | local, 1 |
| Correct! | d, int, | local, 0 |
| Correct! | e, int, | local, 1 |
| Correct! | f, int, | local, 0 |
| Correct! | g, int, | argument, 1 |
| Correct! | z, int, | argument, 1 |

Other Incorrect Match Options:
- this, 2
- local, 4
- this, 1
- local, 3
- this, 0
- argument, 3
- argument, 4
- this, 4
- local, 2
- this, 3

## Question 3

1 / 1 pts

Consider the following **Jack** method:

```
method int blowupstack(int a, int z)    avg
{
        var string message ;
        var int d, c ;

        let c = blowupstack(a,z) ;
        return z ;
}
```

*(handwritten annotations: "1" over a, "2" over z, "avg")*

*(handwritten notes at right:)*
message : local 0
d : local 1
c : local 2

What virtual machine code would implement the assignment statement?

    push argument 1

    push argument 2

    call blowupstack 2

☑ pop local 2

---

    push a

    push z

    call blowupstack 2

☐ pop c

---

    call blowupstack 2

    push argument 1

    push argument 2

☐ pop local 2

---

    call blowupstack 2

    push a

    push z

☐ pop c

call blowupstack 2

push argument 1

push argument 2

☐ pop local 3

---

push argument 1

push argument 2

call blowupstack 2

☐ pop local 3

---

## Question 4

**2 / 2 pts**

Complete the symbol table entries for the variables and parameters in the following **Jack** class:

```
class BankAccount
{
    static int nAccounts ;
    static int bankCommission ;
    field int id ;
    field int balance ;
    field string owner ;          arg 1        2                        3
    method void transfer(int sum, BankAccount from, Date when)
    {
            var int i,j ;   local   0  r1
            var Date due;
                            2
            let balance = (balance + sum) - commission(sum * 5) ;
    }
}
```

The symbol has four columns:

Identifier, type, segment, offset.

| | | |
|---|---|---|
| **Correct!** | nAccounts, int, | static, 0 |
| **Correct!** | bankCommission, int, | static, 1 |
| **Correct!** | id, int, | this, 0 |
| **Correct!** | owner, string, | this, 2 |
| **Correct!** | balance, int, | this, 1 |
| **Correct!** | sum, int, | argument, 1 |
| **Correct!** | from, BankAccount, | argument, 2 |
| **Correct!** | when, Date, | argument, 3 |
| **Correct!** | i, int, | local, 0 |
| **Correct!** | j, int, | local, 1 |
| **Correct!** | due, Date, | local, 2 |

Other Incorrect Match Options:
- static, 3
- this, 4
- local, 4
- static, 4
- local, 3
- static, 2
- argument, 4
- this, 3
- argument, 0