上个quiz一样的题

Category: Jack MyClass.Useful(34,"55")

If you write the following call in a Jack function in class Bob:

MyClass.Useful(34,"55")

but the function MyClass.Useful was declared to take three parameters, an int, an Array and a BankAccount, what happens?

- [ ] The program runs and a segmentation violation occurs when the missing BankAccount parameter is accessed.
- Correct! [x] The program runs but does not work as expected

  The return address will be used as the address of the BankAccount object ...
- [ ] The class Bob does not compile
- Correct! [x] The class Bob compiles and the program assembles correctly.
- [ ] The class Bob compiles but the program does not assemble correctly
- [ ] The program runs and a segmentation violation occurs when "55" is used as an array

Question 2

1 / 1 pts

Category: Compiling Parsing Broken

Which lines of the following Jack class in the file 'CoolClass.jack', contain a syntax error?

```
01 class Broken
02 {
03      int hd ;
04      field Broken tail ;
05
06      int next()
07      {
08            var Bad x ;
09
10            let x = BadClass.returnsnull(tail) ;
11            return x ;
12      }
13 }
```

static class var
field member var

function method constructor

- [ ] 11
- [ ] 10
- Correct! [x] 3

  class variables must be prefixed by the keyword 'static' or 'field'.
- [ ] 9
- [ ] 13

- [ ] 12
- [ ] 4
- [ ] 2
- [ ] 5

Correct!

- [x] 6

Functions must be prefixed by the keyword 'constructor', 'function' or 'method'.
In this case we are using the field tail so this should be prefixed by 'method'.

- [ ] 7
- [ ] 8

Correct!

- [x] 1

The class name must match the filename.

Question 3

1 / 1 pts

Consider these rules for a tokeniser (* means 0 or more):

```
int ::= '0' | (('1'-'9')('0'-'9')*)
name ::= ('a'-'z') ( 'a'-'z'|'0'-'9' )*
```

if starting at "0", then is "0"
"0" or other digits start with "1" – "9"

start with a lowercase letter
followed by any numbers or letter

What tokens would be produced from the following string assuming that whitespace is not returned as tokens?

"43 bobis50 99balloons"

int name int name.

- ( ) int "43", name "bobis", int "50", name "99balloons"
- ( ) int "43", name "bobis50", name "99balloons"

Correct!

- (•) int "43", name "bobis50", int "99", name "balloons"
- ( ) int "43", name "bobis", int "50", int "99", name "balloons"

Question 4

1 / 1 pts

Consider these rules for a tokeniser, * means 0 or more, ? means 0 or 1:

```
num ::= ('0' | (('1'-'9')('0'-'9')*)) ('.' ('0'-'9')*)?
dot ::= '.'
```

How many tokens would be produced from the following string?

"07.09.0.0.0"

"0", "7.09", ".", "0.0", ".", "0"
1 2 3 4 5 6

- ( ) 3

Correct!

- (•) 6
- ( ) 11
- ( ) 7
- ( ) 1
- ( ) 5

The correct answer is 6:

The correct answer is 6:

```
"0"   "7.09"   "."  "0.0"  "."  "0"
```

Question 5

1 / 1 pts

Consider the grammar rule for a Jack local variable declaration:

```
varDec ::= 'var' type varname (',' varname)* ';'
```

var int i,j;

Which of the following code fragments would be able to parse local variable declarations?

tk_var -> pre-defined constant keyword

Correct!

```
mustbe(tk_var) ;
parse_type() ;
parse_varname() ;
while (have(tk_comma))
{
    mustbe(tk_comma) ;
    parse_varname() ;
}
mustbe(tk_semi) ;
```

```
mustbe(tk_var) ;
parse_type() ;
parse_varname() ;
mustbe(tk_semi) ;
```

每次一个 var

```
mustbe(tk_var) ;
mustbe(tk_identifier) ;
mustbe(tk_identifier) ;
while (have(tk_comma))
{
    next_token() ;
    mustbe(tk_identifier) ;
}
mustbe(tk_semi) ;
```

```
mustbe(tk_var) ;
parse_type() ;
parse_varname() ;
mustbe(tk_comma) ;
parse_varname() ;
mustbe(tk_semi) ;
```