"Has a" vs. "Is a" Relationship

Card Game  
| have member |
| Deck       |
| Card       |
| Go Fish    |
| Black Jack |

Paragraph

Design 1st
Software engineering architecture

Board Game
| is a |
| inherited from Board Game |

Paragraph

Member
have words
have characters

Paragraph

Member
have sentences

Odds and Ends...

• Test – Wednesday, 4/29
• Assignment #2
2. `return` object from function `netflix n = fun();`

Definition: `netflix fun (...) {
    netflix n;
    return n;
}`

Copy constructor:

3. `netflix n;
   n = fun();`

Destructor for `n`:

No copy constructor:

`netflix * fun() {
    netflix * n;
    n = new netflix;
    return n;
} NO!!!`

Good:

return new netflix;
③ `netflix n;`  
`netflix n2 = n;`  
↑  
`copy constructor`

VS.  
`netflix n, n2;`  
↑  
`n2 = n;`  
= op overload
Revisit “has a” Relationship

- What is the “has a” relationship?
- How can we make a paragraph class that has strings?
#ifndef PARA_H
#define PARA_H
#include "./mystring.h"

class paragraph {
    public:
        paragraph();
        string get_sentence() const;
        void set_sentence(const string &);
    
    private:
        string sentence; // have a relationship

};

#endif
#include "./paragraph.h"
#include <iostream>

paragraph::paragraph() : sentence("hello")
{
}

paragraph::paragraph() {
    sentence=string("hello");
}

string paragraph::get_sentence() const {
    return sentence;
}

//We'll write set_sentence later


```cpp
#include <stdio.h>
#include "./paragraph.h"

using std::cout;
using std::endl;
using std::fstream;
using std::ios;

int main() {
    paragraph p;
    cout << p.get_sentence().at(1) << endl;

    string str2("hello");
    //string str=str2; //supposed to call copy constructor to make new
    string str;
    str=str2;    //since both created, then call = overload
    fewer lines
}```