



Samueli
Computer Science



CS32: Introduction to Computer Science II

Discussion Week 2

Junheng Hao, Arabelle Siahaan

Apr. 12, 2019

- Course website: <http://web.cs.ucla.edu/classes/spring19/cs32/>
- CS32 Discussion 1F
 - Time and place: 12:00-1:50pm, Friday @Boelter Hall 5264
 - Discussion sessions on week 2-10
- TA: [Junheng Hao](#)
 - Email: haojh.ucla@gmail.com
 - Office Hours: Wednesdays 8:30am-10:30am & Fridays 9:00-10:00am @BH3256S
- LA: Arabelle Siahaan
 - Email: arabellekezia@ucla.edu
 - Office Hours: Tuesdays 1-2 pm & 4-5 pm
- Discussion Website: <https://www.haojunheng.com/teaching/cs32-spring19/>
(Slides will be posted here!)

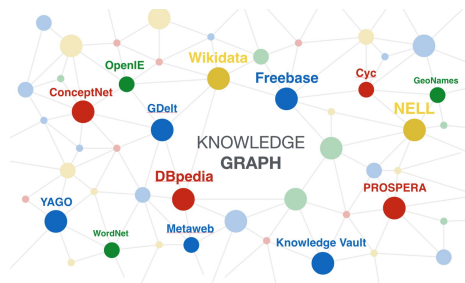
About TA: Junheng Hao

- I am a second-year PhD student at ScAi Data Mining Lab, Department of Computer Science, working with [Yizhou Sun](#) and [Wei Wang](#).
- Other courses I taught: CS145 (Intro to data mining)
- Research Interests:
 - Knowledge Graph & Knowledge Bases
 - Natural Language Processing
 - Graph Mining
 - Healthcare + AI
- I am also learning languages (German & Spanish) for research and for fun!



Taken on August 14, 2017 at
Wadi rum, Jordan

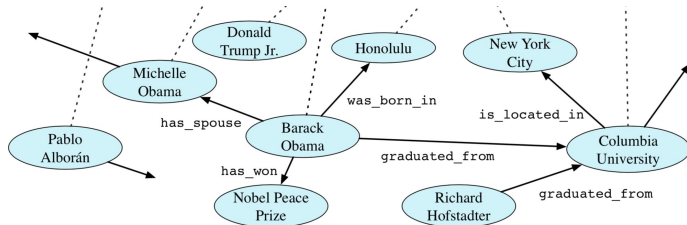
Credit: Bo Chen (UBC)



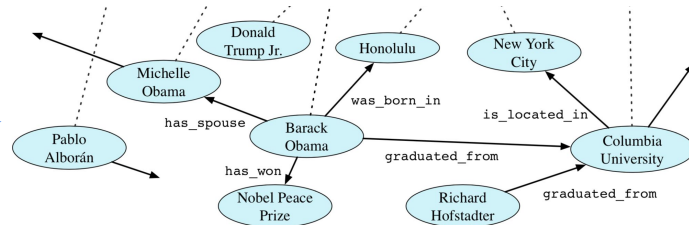
More about my research

→ Knowledge Graphs

English Knowledge Graph



German Knowledge Graph



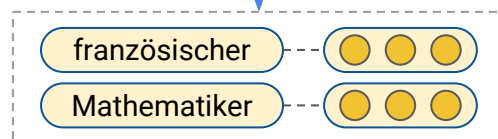
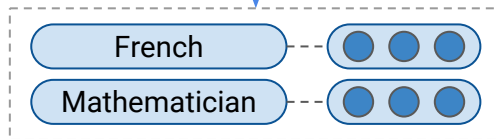
→ Description

French mathematician, currently Professor at the Collège de France, IHÉS, The Ohio State University and Vanderbilt University. He was an Invited Professor at the Conservatoire national des arts et métiers (2000)

Alain Connes

Alain Connes, ist ein französischer Mathematiker und Träger der Fields-Medaille.

→ Word



- Mining knowledge graphs \rightarrow AI + KG + NLP (+ Healthcare)
 - Multi-view knowledge graphs (inferences on Wiki ontology and instances)
 - Multilingual knowledge graphs (cross-lingual search)
 - Healthcare: Automatic analyzer reports via medical knowledge bases
- If you would like to join us in AI & Data Mining group as an undergrad researcher, contact Prof. Yizhou Sun, or Prof. Wei Wang.
 - Definitely, you are also welcomed to our lab in later quarter if you need more time to learn basic machine learning knowledge. :)

- Start your research experiences in AI/Data mining areas **EARLY!**
- Require advisor's approval (Yizhou Sun, Wei Wang, etc). → Send emails!
- Ask yourself:
 - Your background on programming (python preferred), math & statistics (linear algebra, probability, calculus) and machine learning/data mining knowledge
 - Your past project/research experiences
 - How many hours per week (on average) can you contribute to research? Balance your coursework and other activities.
- If allowed by professors, you can come anytime throughout the year, even summer quarters!

What you will learn in this course...

More about C++ and data structures. Well, that's a lot. And very important.

- Review of C++
- **Object-oriented programming:** Data abstraction, Inheritance and Polymorphism, Recursion, ...
- **Data structures:** Arrays, Lists, Trees, Graphs, Hash tables, ...
- **Algorithms** (eg. sorting) **and complexity analysis**

CS32 is the most beneficial course for your future job interview if you apply for Software Development Engineers (SDE).

- Part 1: Review lectures ~50 minutes
 - Review some topics, questions and examples based on last week's lectures
 - Some extended topics from lectures
 - Some typical questions & examples
 - Homework & projects
- Part 2: Exercise ~50 minutes
 - Hands-on exercise questions (from worksheets) in groups (2-3 students per group)

Tell me what you'd like to do in the discussion!

- If you have conceptual questions or problems about lectures, ...
- If you find some problems (not limited in worksheets) are particularly difficult for you, ...
- Then, write emails about it to me with **keyword "CS32-DisQ"** !

- Check course website: <http://web.cs.ucla.edu/classes/spring19/cs32/>
- Homework 1 is due on **Tuesday, April 16.**
- Project 2 is due on Tuesday, April 23.
- Midterm 1 is scheduled on Thursday, April 25.

Important: Special notes about homework and projects!

1. Check projects & homework requirements first! You may get 0 if you do not follow these guidelines and restrictions!
2. When debugging, check your error message! Search Google or Stackoverflow for similar error and find possible reasons.
3. Do **NOT** let you TAs or LAs debug through emails (almost impossible actually)!
4. Others...



- Dynamic memory allocation in C++
- Constructor, destructor, copy constructors, and assignment operators
- Homework 1: Sequence
- Data structure: Arrays & Linked Lists (probably just start)

- Let's first compare with *Static Memory Allocation*!
- If we want to type in a paragraph and save it into a C-string,

```
#define MAXLENGTH 10000  
char s[MAXLENGTH+1];  
cin.getline(s);
```

- What if the paragraph is extremely long? → Out-of-bound
- What if the paragraph has only five words? → Overallocated Memory (Waste)

Dynamic memory allocation

new in C++

What if we want to fit the paragraph into a C-string with right the sufficient size?

```
<type> *<name> = new <type>[<#elements>];  
char *article = new char[length];
```

Unsigned int variable

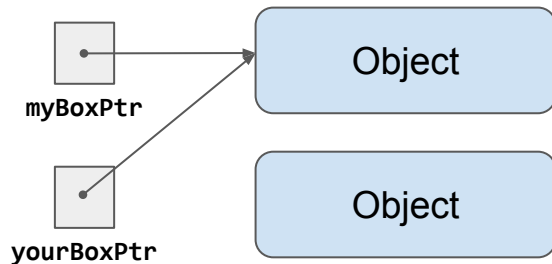
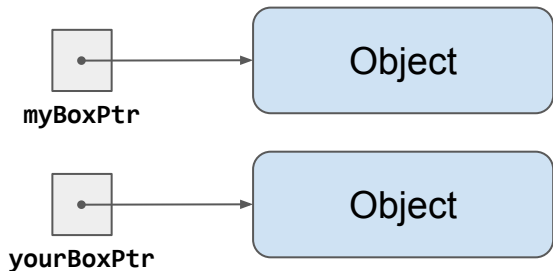
new will dynamically allocate the sequential memory space for the requested data type and size and return the starting address of the allocated memory space.

Variables allocated with **new** will remain in the memory unless we manually delete it which means dynamic allocation has to be deleted once we no longer use it.

```
delete [] article
```

Dynamic memory allocation

Memory Leak



```
// Create first object
MagficBox* myBoxPtr = new MagficBox

// Create second object
MagficBox* yourBoxPtr = new MagficBox

// Assignment causes an inaccessible object
yourBoxPtr = myBoxPtr
```

Dynamic memory allocation

A Problematic Code (1)

```
#include <iostream>
#include <string>
using namespace std;

class Node{
public:
    Node(int id){
        cout << "Constructor:" << id << endl;
        this->id = id;
    }
    ~Node(){
        cout << "Destructor:" << this->id << endl;
    }
private:
    int id;
};

int main(){
    Node node1(31);
    Node* node2 = new Node(32);
    // do something
    return 0;
}
```

Q: What is the output of this simple “node” program?

Constructor:31
Constructor:32
Destructor:31

Node(32) does not deconstruct!

Dynamic memory allocation

A Problematic Code (2)

Let's check it by [Valgrind](#)!

```
#include <iostream>
#include <string>
using namespace std;

class Node{
public:
    Node(int id){
        cout << "Constructor:" << id << endl;
        this->id = id;
    }
    ~Node(){
        cout << "Destructor:" << this->id << endl;
    }
private:
    int id;
};

int main(){
    Node node1(31);
    Node* node2 = new Node(32);
    // do something
    return 0;
}
```

```
jeffhao@jeffhao:~/Desktop/CS32_example/memleak$ valgrind --leak-check=yes node_prog
valgrind: node_prog: command not found
jeffhao@jeffhao:~/Desktop/CS32_example/memleak$ valgrind --leak-check=yes ./node_prog
==10005== Memcheck, a memory error detector
==10005== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==10005== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==10005== Command: ./node_prog
==10005==
Constructor:31
Constructor:32
Destructor:31
==10005==
==10005== HEAP SUMMARY:
==10005==   in use at exit: 72,708 bytes in 2 blocks
==10005==   total heap usage: 3 allocs, 1 frees, 73,732 bytes allocated
==10005==
==10005== 4 bytes in 1 blocks are definitely lost in loss record 1 of 2
==10005==    at 0x4C2E0EF: operator new(unsigned long) (in /usr/lib/valgrind/vgpreload_memcheck-amd64
-linux.so)
==10005==   by 0x400ACA: main (in /home/jeffhao/Desktop/CS32_example/memleak/node_prog)
==10005==
==10005== LEAK SUMMARY:
==10005==   definitely lost: 4 bytes in 1 blocks
==10005==   indirectly lost: 0 bytes in 0 blocks
==10005==   possibly lost: 0 bytes in 0 blocks
==10005==   still reachable: 72,704 bytes in 1 blocks
==10005==   suppressed: 0 bytes in 0 blocks
==10005==
==10005== Reachable blocks (those to which a pointer was found) are not shown.
==10005== To see them, rerun with: --leak-check=full --show-leak-kinds=all
==10005==
==10005== For counts of detected and suppressed errors, rerun with: -v
==10005== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```


Dynamic memory allocation

Good memory example from textbook

```
/** @file GoodMemory.cpp */
# include "GoodMemory.h"

GoodMemory::GoodMemory() : someBoxPtr(nullptr)
{
} // end default constructor

GoodMemory::~~GoodMemory()
{
    delete someBoxPtr;
} // end constructor

void GoodMemory::unleakyMethod(const double& someItem)
{
    someBoxPtr = new ToyBox<double>();
    someBoxPtr->setItem(someItem)
} // end unleakyMethod
```

Construction & Destruction

Class Composition

Class composition is when a class contains one or more member variables that are objects.

Order of construction:

- Member variables are constructed in order.
- Then the current class constructor is executed.

Order of destruction:

- The current class destructor is executed first.
- The member variables are destructed in the reverse order.

Summary:

- Constructor: Inside -> Outside (In-order)
- Destructor: Outside -> Inside (Reverse Order)

Question: For classes containing members of a class type, what is the order of construction and destruction?

What is the output?

Answer:

A(444)

A(888)

B()

~B()

~A(888)

~A(444)

```
#include <iostream>
#include <string>
using namespace std;

class A
{
public:
    A(){cout << "A()" << endl;}
    A(int x){cout << "A(" << x << ")" << endl; this->id = x;}
    ~A(){cout << "~A(" << this->id << ")" << endl;}
private:
    int id;
};

class B
{
public:
    B():a1(888),a2(444){cout << "B()" << endl;}
    ~B(){cout << "~B()" << endl;}
private:
    A a2;
    A a1;
};

int main()
{
    B b;
    return 0;
}
```

Construction & Destruction

Note: There is a difference between class composition and class inheritance. → Will be explained in later lectures.

```
#include <iostream>
#include <string>
using namespace std;

class A
{
public:
    A(){cout << "A()" << endl;}
    A(int x){cout << "A(" << x << ")" << endl;}
    ~A(){cout << "~A()" << endl;}
};

class B
{
public:
    B():a1(1),a2(2){cout << "B()" << endl;}
    ~B(){cout << "~B()" << endl;}
private:
    A a2;
    A a1;
};

int main()
{
    B b;
    return 0;
}
```

```
#include <iostream>
#include <string>
using namespace std;

class A
{
public:
    A(){cout << "A()" << endl;}
    A(int x){cout << "A(" << x << ")" << endl;}
    ~A(){cout << "~A()" << endl;}
};

class B : public A
{
public:
    B():a1(1),a2(2){cout << "B()" << endl;}
    ~B(){cout << "~B()" << endl;}
private:
    A a2;
    A a1;
};

int main()
{
    B b;
    return 0;
}
```

Copy Constructors

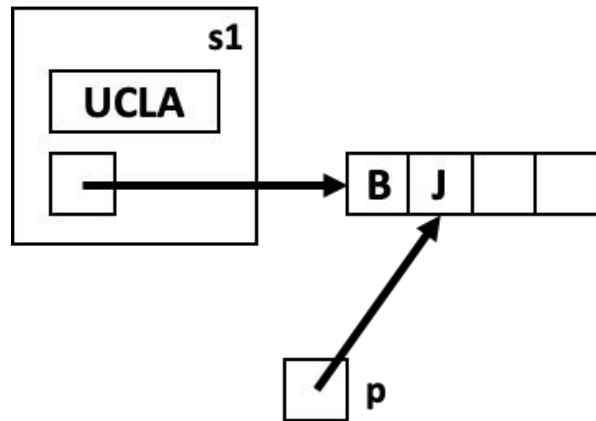
Motivation

Consider an example of UCLA and its students:

```
Student st1("Brian");  
Student st2("John");  
School s1("UCLA");  
s1.addStudent(st1);  
s1.addStudent(st2);  
Student *p = s1.getStudent("John");
```

We want to create a new School called s2, with exactly the same content as s1. In other words, we want to clone s1.

```
School s2(""); s2=s1; // Is this correct?
```



Copy Constructors

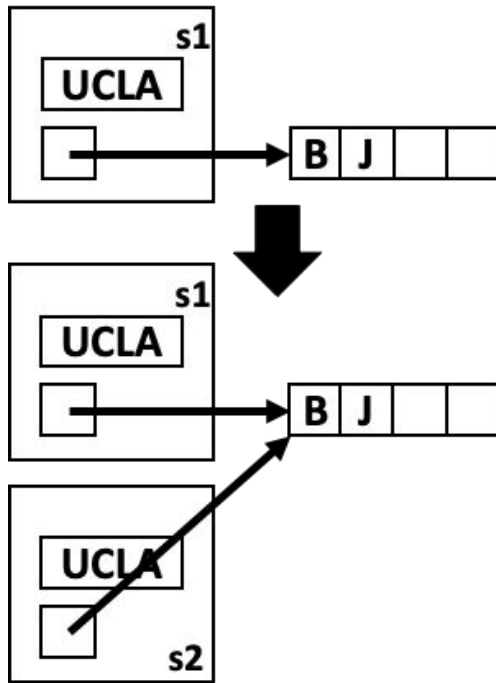
Shallow copy problem

We want to create a new School called s2, with exactly the same content as s1. In other words, we want to clone s1.

```
School s2(""); s2=s1; // Definitely not!
```

What if grab values out of s1 and manually copy them into s2?

```
School s2("");  
s2.setName(s1.getName( ));  
... // copy all members and properties
```



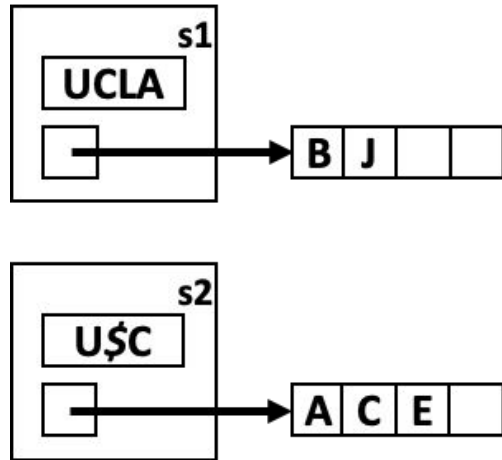
Copy Constructors

Example

Immutable. We don't change the object we're copying from.

```
School::School(const School &aSchool){  
    m_name = aSchool.m_name;  
    m_numStudents = aSchool.m_numStudents;  
    m_students = new Students[m_numStudents];  
    for (int i = 0; i < m_numStudents; ++i)  
        m_students[i] = aSchool.m_students[i];  
}
```

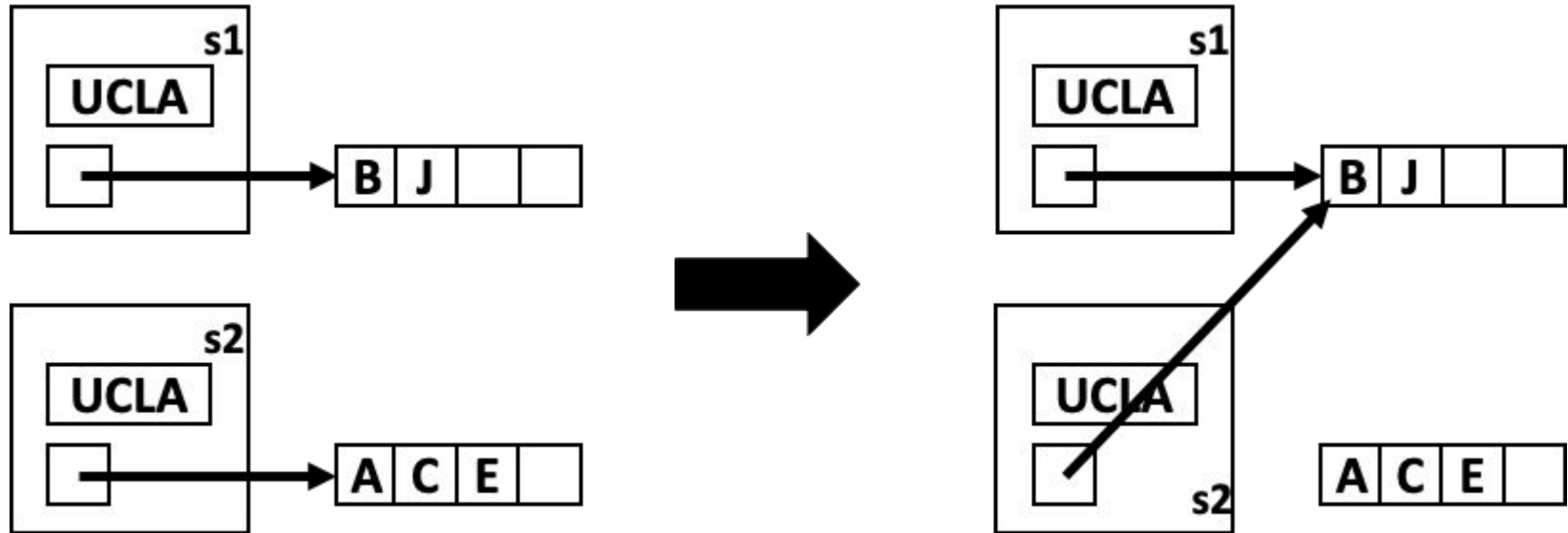
Now you can do: `School s2(s1);`



Assignment Operator

Motivation

What if $s2 = s1$?




Assignment Operator

Example

Assignment operator

```
School& School:: operator=(const School &aSchool){  
    m_name = aSchool.m_name;  
    m_numStudents = aSchool.m_numStudents;  
    m_students = new Students[m_numStudents];  
    for (int i = 0; i < m_numStudents; i++)  
        m_students[i] = aSchool.m_students[i];  
  
    return *this;  
}
```



*Do not forget *this!*

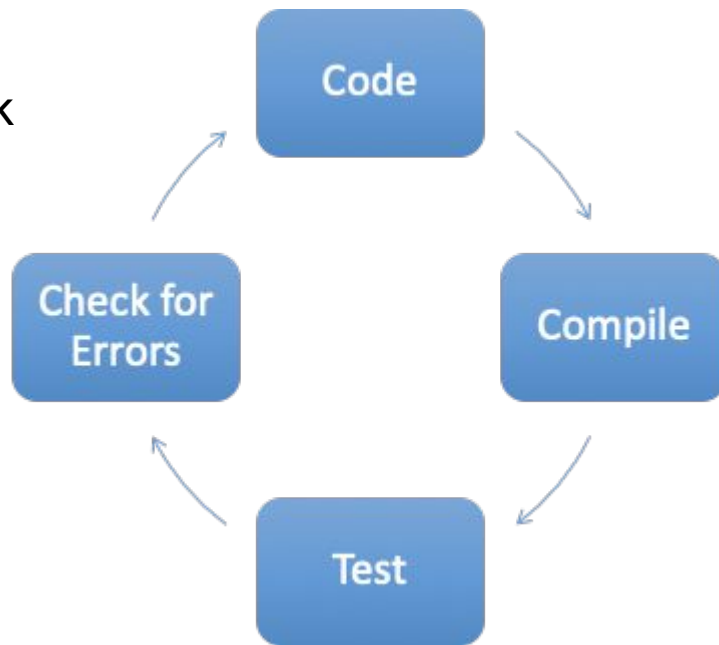
Assignment Operator

Example

Make it better:

```
School& School:: operator=(const School &aSchool){  
    if (this != &aSchool)  
    {  
        m_name = aSchool.m_name;  
        m_numStudents = aSchool.m_numStudents;  
        delete[] m_students;  
        m_students = new Students[m_numStudents];  
        for (int i = 0; i < m_numStudents; i++)  
            m_students[i] = aSchool.m_students[i];  
    }  
    return *this;  
}
```

- Pointers → Memory layout
- Assert
- Debug → Breakpoint, trace, variables, stack
- Pseudocode
- Others...



Before we talk about doing things arrays, let's talk about data structure first!

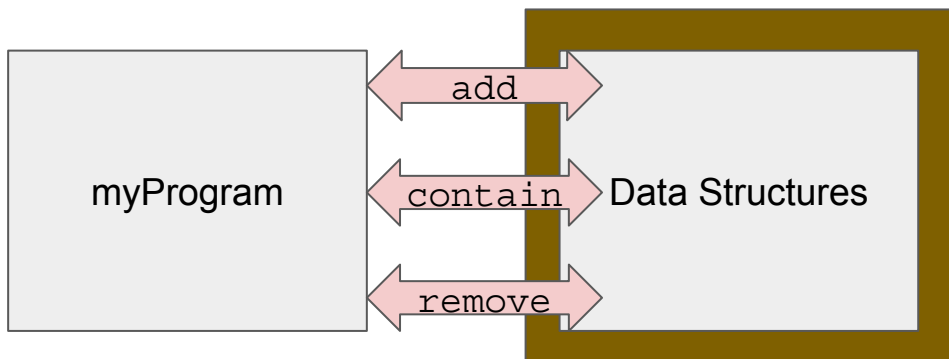
CS32 is not just to write code and run. It is more about organizing data. We call an organization scheme a data structure. For every structure, we define:

- How to store/organize the data items?
- Method to add new data / remove data
- Apply functions on data: most importantly, how to search data?

We also need to know pros and cons of each data structure as well as its efficiency or complexity of algorithms with these data structures.

Arrays (as an example)

- Most basic data structure and most important!



- Add / Contain / Remove → Complexity?
- More implementations: `getIndexOf`, `getFrequencyOf`, ...
- More questions: Using array to implement Linked List? How to sort a array or maintain a sorted array?

Hints for Homework 1 (Q1-3)

Homework 1: Use array to implement Sequence (of strings, integers, etc).

```
// Sequence interface
class Sequence
{
    public:
        Sequence();
        bool empty();
        int size();
        int insert(int pos, const std::string& value);
        int insert(const std::string& value);
        bool erase(int pos);
        int remove(const std::string& value);
        bool get(int pos, std::string& value);
        bool set(int pos, const std::string& value);
        int find(const std::string& value);
        void swap(Sequence& other);
    private:
        ...
}
```

Questions:

1. What data members do you need as private member (or member functions)?
2. Determine which member functions are should be const member functions? (Q1)
3. How to use a type alias ItemType? (Q2)
4. Inline and non-inline functions, order of implementation. (Q3)

Note: You cannot add, change or delete public data or function members! You may use dump functions for testing

5. Requirement of specific functions (like swap).

Hints for Homework 1 (Q4)

Homework 1: Use Sequence to implement ScoreList

```
// Sequence interface
```

```
class ScoreList
{
    public:
        ScoreList();
        bool add(unsigned long score);
        bool remove(unsigned long score);
        int size() const;
        unsigned long minimum() const;
        unsigned long maximum() const;
    private:
        // Some of your code goes here.
};
```

Questions:

1. Keep in mind that your Sequence cannot change from Question 3! Note that Item Type is unsigned long.
2. The words for and while must not appear except in the implementations of ScoreList::minimum() and ScoreList::maximum(). → To use Sequence's interface functions!

Hints for Homework 1 (Q5)

Homework 1: Dynamically allocated array for your Sequence!

```
// Sequence OLD interface
class Sequence
{
public:
    Sequence();
    bool empty();
    int size();
    int insert(int pos, const std::string& value);
    int insert(const std::string& value);
    //same erase, remove, get and set functions
    int find(const std::string& value);
    void swap(Sequence& other);
    // New functions here!
private:
    ...
}
```

Should pass size parameter!

*Add destructor, copy constructor,
and assignment operator!*

Questions:

1. Again, what data members do you need as private member (or member functions)?
2. How to construct dynamically allocated array?
3. Functions to be added:

~Sequence();
Sequence(const Sequence& other)
Sequence& operator=(const Sequence& other)

Hints for Homework 1

One final note when you are working on Homework 1: Do some testing!

```
// Example: to test remove function
```

```
Sequence s;  
s.insert(0, "a");  
s.insert(1, "b");  
s.insert(2, "c");  
s.insert(3, "b");  
s.insert(4, "e");  
assert(s.remove("b") == 2);  
assert(s.size() == 3);  
string x;  
assert(s.get(0, x) && x == "a");  
assert(s.get(1, x) && x == "c");  
assert(s.get(2, x) && x == "e");
```



Make sure that you follow the requirements of “Turning it in”!



Samueli
Computer Science



Break Time! (5 minutes)

Q & A

About LA: Arabelle Siahaan

- I am a second year Computer Science student from Indonesia :)
- I have LA-ed for CS 31 and CS 32 (this is my second time LA-ing for CS32!)
- CS classes I've taken:
 - CS 31
 - CS 32
 - CS 33
 - CS 35L
 - CS 111

(You can ask me questions about these classes!)

Group Exercises: Worksheet 1

- Exercise problems from **Worksheet 1** (see “LA worksheet” tab in CS32 website). Answers will be posted after all discussions.
- Our LA Arabelle will help you go through these problems.
- Worksheet is optional. We encourage you to solve the problem without working on a computer.



Samueli
Computer Science



Thank you!

Q & A