CSci 1113: Introduction to C/C++
Programming for Scientists and Engineers
Homework 8
Spring 2020

Due Date: Friday, April 24, 2020 before 11:55pm.

Instructions: This is an individual homework assignment. There are two problem worth 20 points each. Solve the problem below by yourself (unlike the labs, where you work collaboratively), and submit the solution as a C++ source code file. Here are a few more important details:

1. Unlike the computer lab exercises, this **is not** a collaborative assignment. You must design, implement, and test the solution to each problem on your own without the assistance of anyone other than the course instructor or TAs. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class: examples from the textbook, lectures, or code you and your partner write to solve lab problems. Otherwise obtaining or providing solutions to any homework problems for this class is considered academic misconduct. See the "collaboration rules" file on the class website page for more details, and ask the instructor if you have questions.

2. Because all homework assignments are submitted and tested electronically, the following are important:
   - You follow the naming conventions mentioned at the end of the problems.
   - You submit the correct file(s) on gradescope (https://www.gradescope.com/) by the due deadline.
   - You follow the example input and output formats exactly given in each problem description.
   - **Regardless of how or where you develop your solutions, your programs compile and execute on gradescope computers (which run Linux/Ubuntu operating system like the cselabs machines).**

3. The problem descriptions will usually show at least one test case and the resulting correct output. However, you should test your program on other test cases (that you make up) as well. Making up good test cases is a valuable programming skill, and is part of ensuring your code solution is correct.

Problem A: Adding trains (20 points)
Start with the "itsABirdItsAPlane...ItsATrain.cpp" file on the website:
http://www-users.cselabs.umn.edu/classes/Spring-2020/csci1113/assignments/itsABirdItsAPlane...ItsATrain.cpp

Fill in the missing functions/classes to enable the program to compile and work correctly. This program simulates a train, where you start with just an engine. You have three options: go to the next train (forward towards the engine), go to the previous train (away from the engine) or add a train behind the current train car. **You may not change main(), we will check to ensure it is exactly the same.** You should be able to train cars anywhere, and this will insert it into that part of the train. **For this part, you do not need to worry about deleting dynamic memory**.

Hint: when making the add() function, it would probably help to draw it out and figure out how many arrows/pointers you need to change.

Example 1 (user input is underlined):

```
Current train: Engine
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
a
Which train is this?
4
Current train: Engine
Previous train: 4
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
a
Which train is this?
1
Current train: Engine
Previous train: 1
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
p
Next train: Engine
Current train: 1
Previous train: 4
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
a
Which train is this?
3
Next train: Engine
Current train: 1
Previous train: 3
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
a
Which train is this?
2
Next train: Engine
Current train: 1
Previous train: 2
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
p
Next train: 1
Current train: 2
Previous train: 3
```

```
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
p
Next train: 2
Current train: 3
Previous train: 4
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
p
Next train: 3
Current train: 4
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
p
Next train: 3
Current train: 4
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
a
Which train is this?
5
Next train: 3
Current train: 4
Previous train: 5
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
p
Next train: 4
Current train: 5
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
n
Next train: 3
Current train: 4
Previous train: 5
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
n
Next train: 2
Current train: 3
Previous train: 4
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
n
Next train: 1
Current train: 2
Previous train: 3
```

```
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
n
Next train: Engine
Current train: 1
Previous train: 2
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
n
Current train: Engine
Previous train: 1
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, or (q)uit?
q
```

When you are done, name the source code file hw8A.cpp. Then log into gradescope and upload your file for the "Homework 8A" submission. **If you name your file incorrectly it will be unable to compile and run your code, so you will fail all test cases**. You may submit cpp files as many times as you want until the deadline to try and fix the code if you fail a test case. Following rigorous naming conventions and using test cases are something computer programmers often must do in "real life" programming , and so submitting your program with the correct name and functionality is part of doing this assignment correctly.

Problem B: Removing trains (20 points)
Build off your answer for problem A (keep the old functionality). Add new functionality to detach the previous train (the one behind the current train), if it exists (if there is no "previous train" simply do nothing). **This time you need to ensure there are no memory leaks and all "new"s are deleted.** (This time you will need to modify main to add this option). Any cars on the train that are left once the user quits the loop should be deleted at the end of main().

See the very bottom for a way to test memory leaks on a cselabs machine or through vole.

Example 1 (user input is underlined):

```
Current train: Engine
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
a
Which train is this?
3
Current train: Engine
Previous train: 3
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
a
Which train is this?
1
Current train: Engine
Previous train: 1
```

```
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
p
Next train: Engine
Current train: 1
Previous train: 3
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
a
Which train is this?
TERIBAD
Next train: Engine
Current train: 1
Previous train: TERIBAD
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
a
Which train is this?
2
Next train: Engine
Current train: 1
Previous train: 2
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
p
Next train: 1
Current train: 2
Previous train: TERIBAD
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
d
Next train: 1
Current train: 2
Previous train: 3
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
d
Next train: 1
Current train: 2
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
d
Next train: 1
Current train: 2
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
n
```

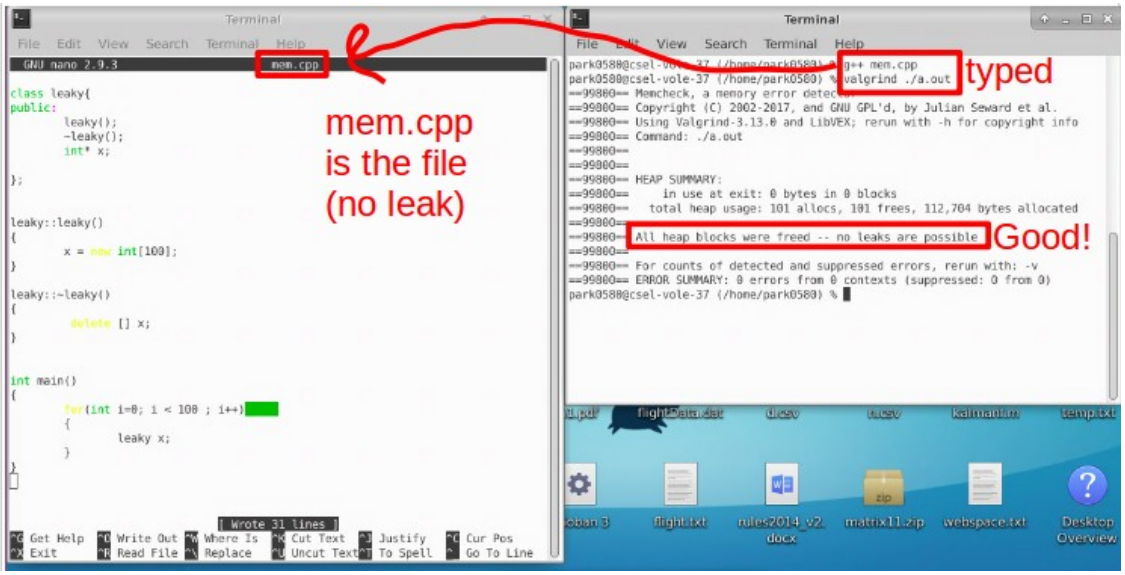
```
Next train: Engine
Current train: 1
Previous train: 2
Do you wish to go to the (n)ext train, (p)revious train, (a)dd a
train, (d)etach a train, or (q)uit?
q
```

When you are done, name the source code file hw8B.cpp. Then log into gradescope and upload your file for the "Homework 8B" submission. **If you name your file incorrectly it will be unable to compile and run your code, so you will fail all test cases**. You may submit cpp files as many times as you want until the deadline to try and fix the code if you fail a test case. Following rigorous naming conventions and using test cases are something computer programmers often must do in "real life" programming, and so submitting your program with the correct name and functionality is part of doing this assignment correctly.

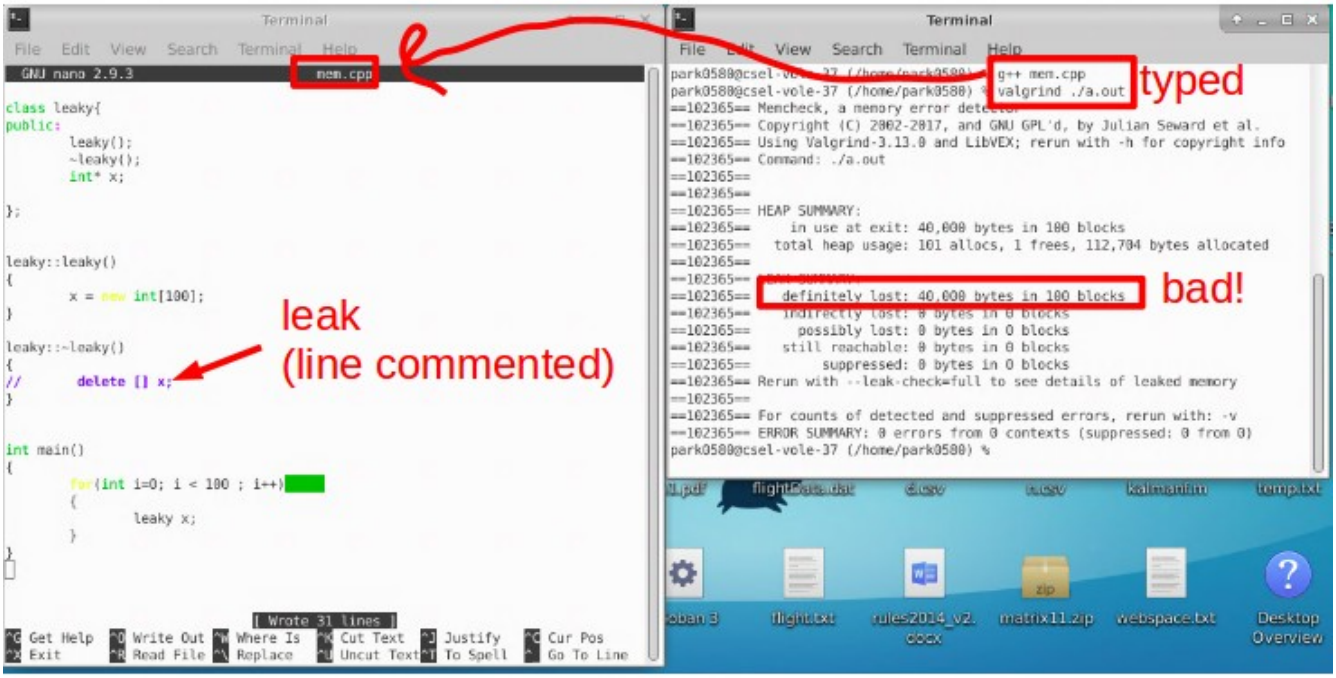
Testing for memory leaks

Both vole and the cselabs machine have a program called "valgrind" that gives you information on memory leaks. It is probably easiest to use a terminal to "cd" to the directory where the code is (or navigate there in the gui and right-click -> "Open Teriminal").

Once there, compile the code with the "g++" command then run "valgrind ./a.out" as shown below:



If you have a memory leak of some sort, it should look more like this:


Terminal
File Edit View Search Terminal Help
GNU nano 2.9.3 mem.cpp
class leaky{
public:
leaky();
~leaky();
int* x;
};
leaky::leaky()
{
x = new int[100];
}
leaky::~leaky()
{
// delete [] x;
}
int main()
{
for(int i=0; i < 100 ; i++)
{
leaky x;
}
}
[Wrote 31 lines]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
leak
(line commented)
Terminal
File Edit View Search Terminal Help
g++ mem.cpp
valgrind ./a.out
typed
==102365== Memcheck, a memory error detector
==102365== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==102365== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==102365== Command: ./a.out
==102365==
==102365==
==102365== HEAP SUMMARY:
==102365== in use at exit: 40,000 bytes in 100 blocks
==102365== total heap usage: 101 allocs, 1 frees, 112,704 bytes allocated
==102365==
==102365== LEAK SUMMARY:
==102365== definitely lost: 40,000 bytes in 100 blocks
==102365== indirectly lost: 0 bytes in 0 blocks
==102365== possibly lost: 0 bytes in 0 blocks
==102365== still reachable: 0 bytes in 0 blocks
==102365== suppressed: 0 bytes in 0 blocks
==102365== Rerun with --leak-check=full to see details of leaked memory
==102365==
==102365== For counts of detected and suppressed errors, rerun with: -v
==102365== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
park0580@csel-vole-37 (/home/park0580) %
bad!
flightData.dat
kalmanfilm
temp.txt
flight.txt
matrix11.zip
webspace.txt
Desktop Overview