D00 – Linked Lists

Student Information

Integrity Policy: All university integrity and class syllabus policies have been followed. I have neither given, nor received, nor have I tolerated others' use of unauthorized aid.

I understand and followed these policies: ◯ Yes ◯ No

Name:

Date:

Submission Details

Final ***Changelist*** number:

Verified build: ◯ Yes ◯ No

Required Configurations: DEBUG only

Test Passed:

Discussion (What did you learn):

Verify Builds

- Follow the Piazza procedure on submission
  - Verify your submission compiles and works at the changelist number.
- Verify that only MINIMUM files are submitted
  - No – Generated files
    - *.pdb, *.suo, *.sdf, *.user, *.obj, *.exe, *.log, *.pdb, *.db, *.user
    - Anything that is generated by the compiler should not be included
  - No – Generated directories
    - /Debug, /Release, /Log, /ipch, /.vs
- Typical files project files that are required
  - *.sln, *.csproj, *.cs,
  - App.config, AssemblyInfo.cs, CleanMe.bat
  - Resources Directory:
    - *.tga, *.dll, *.wav, *.glsl, *.azul

Standard Rules

Submit multiple times to Perforce

- Submit your work as you go to perforce several times
  - Design Patterns – no minimum
  - Sprints – minimum of 5 real submissions (generally 10+)
    - As soon as you get something working, submit to perforce
    - Have reasonable check-in comments
      - Points will be deducted if minimum is not reached

Submission Report

- Fill out the submission Report
  - No report, no grade

Code and project need to compile and run

- Make sure that your program compiles and runs
  - Warning level 4
  - NO Warnings or ERRORS
    - Your code should be squeaky clean.
  - Code needs to work “as-is”.
    - No modifications to files or deleting files necessary to compile or run.
  - All your code must compile from perforce with no modifications.
    - Otherwise it's a 0, no exceptions

Project needs to run to completion

- If it crashes for any reason…
  - It will not be graded and you get a 0

No Containers

- No Containers or Collections
  - List, maps, trees, array lists, list, queues, stacks
- No Template or generic parameters
- No arrays
  - You need to do this the old fashion way - ***YOU EARNED IT***
    - No ARRAYs – hard requirement
    - Nothing should use the [ ] operator in your code

Leave Project Settings

- Do NOT change the project or warning level
  - Any changing of level or suppression of warnings is an integrity issue
  - Do not add or link any external libraries not explicitly given

Simple C#

- No .Net
- We are using the basics C#
  - Types:
    - Class, Structs, intrinsic types (int, float, bool, etc…)
  - Basics language features
    - Inheritance, methods, abstract, virtual, etc…
  - No properties – set/get generated accessors

No Debug code or files disabled

- Make sure the program has only active code
  - If you added debug code or commented out code,
    - please return to code to active state or remove it

Due Dates

- See Piazza for due date and time
- Submit program perforce in your student directory assignment supplied.
- Fill out your this ***Submission Report*** and commit to perforce
  - ***ONLY*** use Adobe Reader to fill out form, all others will be rejected.
  - Fill out the form and discussion for full credit.

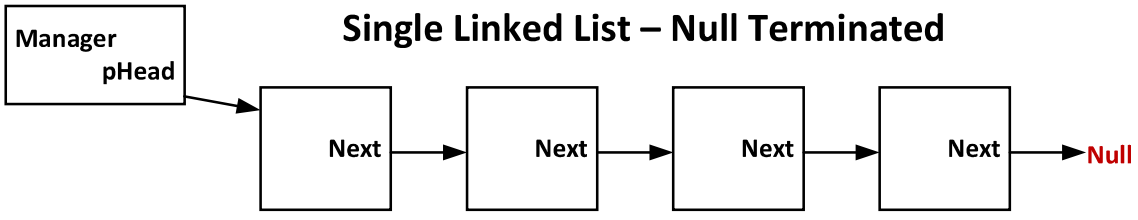
Goals

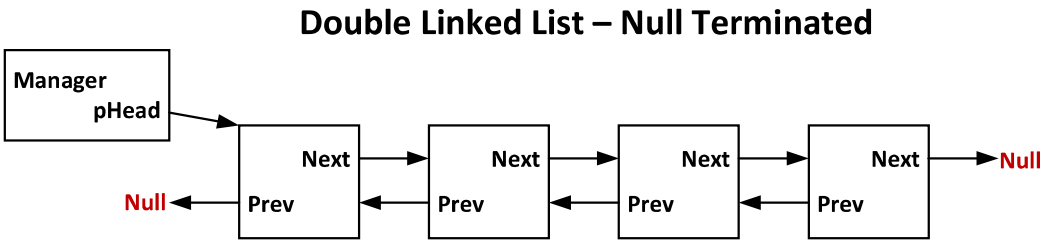
- Learn
  - Linked Lists
    - Double Linked Lists
      - Null Terminated
      - Circular
      - End Pointer
    - Single Linked List
      - Null Terminated

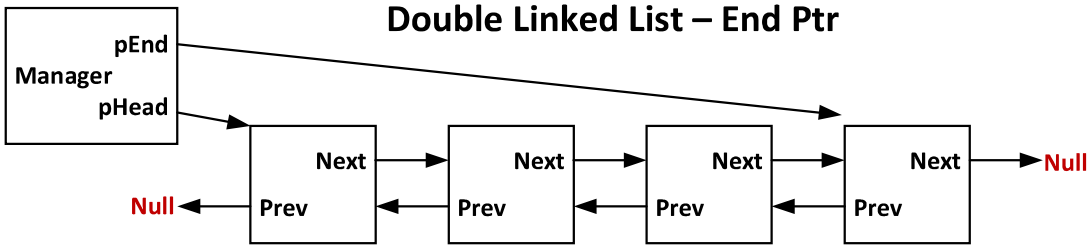
Assignments

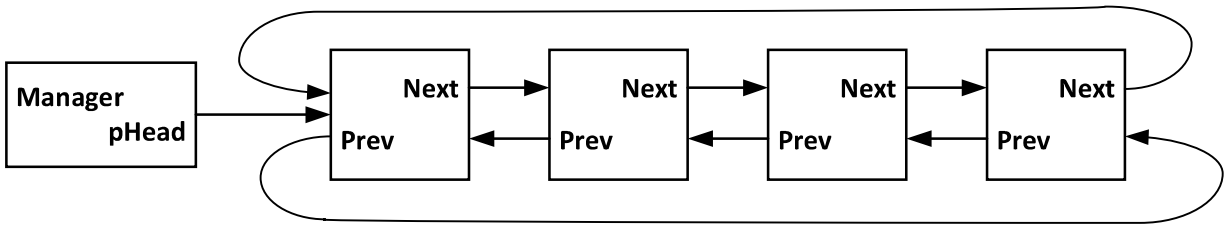
General:

- Linked link comparisons
  - Warm-up and coding exercise
  - Next week we will cut and paste this code into our sprint
- Implement 4 methods for each type of linked lists
  - Add to Front()
    - Insert a node to the front of the list
  - Add to End()
    - Insert a node to the end of the list
  - Remove()
    - Remove any given node on the list
    - Assume the node is valid and exists
  - RemoveFromFront()
    - Remove and return the first node on the list
- Do not change the class layout or data for the supplied types









General guidelines:

- Idea is to get you comfortable with these patterns
  - You will include these concepts into the Space Invaders project
- Create UML diagrams to help
  - Post on Piazza questions and clarifications
- No need to add any files… the unit tests are fully stubbed out

Make sure you delete these using directives (we are not using them)

- `using System.Collections.Generic;`
- `using System.Linq;`
- `using System.Text;`
- `using System.Threading.Tasks;`

Development

- Store project in student directory in perforce
- Do your work in the supplied project

Submission

- Do not add any files to this assignment
  - Just check out from Depot… modify code… submit
- Submit your WORK into the supplied directory into perforce:
  - /student/<yourname>/D01_Singleton/... (for example)
    - You need to submit a complete C# project
  - Solution, project and C# files (whatever it takes to build the project)
    - Do not submit anything that is auto generated
  - Run the supplied CleanMe.bat before submission
    - Should cleanup files
- Fill out the Submission report and submit that pdf to your student directory

Validation

Simple checklist to make sure that everything is submitted correctly

- Is the project compiling and running without any errors or warnings?
- Does the project runs ***ALL*** without crashing?
- Is the submission report filled in and submitted to perforce?
- Follow the verification process for perforce
  - Is all the code there and compiles "as-is"?
  - No extra files

Hints

Most assignments will have hints in a section like this.

- Linked List review
  - You might need to brush up on your data structures
  - Give this assignment a try without references
    - You can do it

Troubleshooting

- Debugging links can be hard
  - Use prints that display the name to help you see the connections
  - Included is a method call Dump()
    - Prints or Dumps info to the screen
    - Its what I used
  - HashCode are unique identifiers – they might also help instead of "pointers"
- Print, print, print
  - Draw diagrams to help you understand
- Have fun… this shouldn't be stressful
  - Slow and steady discovery and development will get you there.
  - Its not hard… just different way of solving problems