Assignment 7 – April 29, 2021 at 11:59pm

This assignment may be done in pairs.

If you want a partner, but don’t have one yet, post/respond to a partner request in the discussion board. Please edit your message when you have found a partner to indicate you are no longer looking.

**\*\*\* NO LATE WORK IS ACCEPTED ON THIS ASSIGNMENT. ALL WORK MUST BE TURNED IN BEFORE DEAD DAY (APRIL 30) – THEREFORE, ALL WORK IS DUE AT 11:59PM ON APRIL 29 \*\*\***

# Objective

The goal of this assignment is to practice using and implementing classes in C++. In addition, you will practice interacting with a base class as a member variable within the class you are implementing. This program will mimic the NFL Draft selection process, where we are using a Draft class, and the class itself uses the Player class. You’ll be writing the interface in Draft.h and then implementing the member methods in Draft.cpp. **You will use the starter code provided to start development.**

# Description

After a record-setting year of profits for her farm, Farmer Jessie wanted to trace her ancestry to see if she has been the most successful farmer in her lineage. Imagine her surprise when she discovered that her ancestor, Giovanni, played in the NFL! She decided to go a different direction with her research, and with the help of Youngster James, they wanted to explore how the NFL draft worked and write a program to illustrate it better. But they need your help!

For this assignment, we’ll be mimicking the NFL Draft! Teams will be able to choose one player per round in a total of 3 rounds. There will be 4 teams participating and we will be choosing from a list of players provided by a text file (PlayerInfo.txt).

In order to have teams choose a player, the tasks for this assignment are to implement methods within the **Draft** class. The **Draft** class has *print* methods, *validation* methods, *accessors (getters)*, and *mutators (setters)*. There’s also a helper function that does the actual drafting of players.

The **Draft** class has several private member variables. *MAX\_PLAYERS* is a static const variable that indicates the maximum number of players will be 100 in our allPlayers[] array. This array is full of **Player** objects from the **Player** class. Next, there are two private integers, numPlayers (how many total players have been read in) and numAvailable (how many players are still available to be chosen in the draft).

You are allowed to work with a partner for this assignment. You must include ***both*** you and your partner’s names on the report, in the comments at the top of the code, and you will print out both of your names and IDs at the beginning of your program’s execution. Only one of you should turn in the code and the report through Blackboard, **with a comment in the text submission area stating with whom you worked.** The other person should just submit a comment in the text submission area stating their partner’s name.

**Before you get started you must download and use the starter code for this project.**

## Approach

To begin, run the starter code provided to you and ensure it works as expected. It should compile, display a welcome message, and begin the three rounds. Right now, none of the choices will work correctly, but if you choose 3, 4, 5, or 6, then you will move onto the next round. At the end of each of the rounds, the user will see all the players that were added to their team (should also be empty as of right now).

You will be implementing **13** methods for a total of 16 methods in the **Draft** class. The default constructor, destructor, and a print method have been provided to you to get you started. The functions are described in the starter code and will be further described later in this document. You will receive more credit for a subset of methods that work correctly than code which attempts all methods but is poorly implemented.

After **each** method is implemented, you should save the file and quit. Then go into the main function and uncomment the portion of main that tests that particular method (this is specified for each function for your convenience!). Compile and run tests to ensure that method works properly, as well as ensuring that this newly implemented method doesn’t wreak havoc on any of the other methods you’ve implemented. It’s a good idea to try a variety of test cases, including invalid inputs so you can test if your error checking works. This is called **incremental development** and is one of the strongest skills of good developers. See below for descriptions of each method and how it is suggested to approach the problem.

As stated, you will receive more credit for completing a subset of methods correctly than merely attempting all methods incorrectly. Each method that is correctly implemented will be worth 4 points for a total of 52 points of correctness. The remaining 48 points of your 100 point assignment is split amongst style (comments and formatting), design decisions (i.e. variable names), testing (typescript), and documentation (report).

# Description of Required Methods in Draft & Suggested Approach

You should start with the copy constructor and destructor. With the copy constructor (method b), it should set the number-of-total-players and number-of-available-players variables in this object equal to the number of player/number available variables of the passed in Draft object. It should also go through the array of Player objects and copy those objects from the Draft object into this object’s array. With the destructor (method c), nothing should be implemented here.

## Default Constructor [given]

*Parameters*: none

Initialize all of the attributes (or instance variables) of the class: initialize the number-of-total-players, and number-of-available-players to zero. This is provided to you! ☺

## Copy Constructor

*Parameters*: Draft object

This function will set the number-of-total-players and number-of-available-players in the current instance of the class equal to those attributes of the Draft object which is passed in. It will also iterate through the array of Player objects from the passed-in Draft object and copy them into our current instance.

*Uncomment*: Draft.h – line 21

## Destructor [given]

*Parameters*: none

Since there is no dynamic memory associated with this class, the destructor will not have anything in it. However, good coding convention dictates that we still include it to get in the habit for bigger projects where we do need to deallocate the memory once we’re done with it so that we don’t end up with a memory leak! This is provided to you! ☺

When you have completed those (including going into Draft.h and uncommenting out the method header for the copy constructor as instructed), test your code and make sure it still behaves as before.

## printAllPlayers [given]

*Parameters*: none

*Returns*: nothing

Iterate through the array of allPlayers[] and print them to the screen. You will use this method after reading in the file to show that your readFile method works as expected. This method is provided to you! ☺

## readFile

*Parameters*: a string that contains the filename that will be read

*Returns*: nothing

Open the file with the pass-by-value string filename and start populating the Player array. You will read in firstName (string), lastName (string), position (string), overallRank (integer), and positionalRank (integer). If the read succeeds, add them into the Player array. [Remember that we already have an array of Player objects, so you’ll want to use the setters to set the current Player’s information to what you just read in.] Be sure to set them also as untaken (i.e. available to be chosen in the draft). Recall that the number-of-total-players represents the total number that have been read in, and number-of-available-players represents the number who are available to be drafted. When you have finished reading in the file, these two numbers should both be the same. You will also want to call printAllPlayers (method d) to show that the file is read in correctly. Be sure to look at which methods are provided to you in the Player.cpp file!

*Uncomment*: Draft.h – line 29, hw7.cpp – line 132 (where readFile is called)

*Testing*: Now the list of Players should print to the screen upon running the program

Test your code again! Does it still run like you expect? You’ve now loaded your text file of Players and they should be printed to the screen. Let’s work on seeing what’s happening next.

## getNumPlayers

*Parameters*: none

*Returns*: an integer of the number of total players that have entered the draft

This method is our accessor (or getter) for our private variable “numPlayers” in the Draft class. It will return the number of players that have entered the draft (i.e. the number of players that were read in from the PlayerInfo.txt file). This method can be implemented with a single return statement.

*Uncomment*: Draft.h – line 25, hw7.cpp – line 136

*Testing*: When you run this function, you should see a line right after “Welcome to the 2021NFL Draft!” that states there are 25 players available (if you are using the provided PlayerInfo.txt file).

## getNumAvailable

*Parameters*: none

*Returns*: an integer of the number of players who are still available to be drafted

This method is our accessor (or getter) for our private variable “numAvailable” in the Draft class. It will return the number of players who have not been drafted to a team yet. This method is tested at the very end of our code, but we can go ahead and uncomment it now! This method can be implemented with a single return statement.

*Uncomment*: Draft.h – line 26, hw7.cpp – line 162

*Testing*: At the end of the program run, you should see the number of remaining players. When we first start out (and haven’t drafted anyone yet), this should say that there are 25 players remaining (if you are using the provided PlayerInfo.txt file).

## printAvailablePlayers

*Parameters*: none

*Returns*: nothing

Iterate through the array of allPlayers[] and print them to the screen if they are not currently taken in the draft. Print the number of available players at the beginning of this method so you can easily check if you’ve drafted them correctly in a little while.

*Uncomment*: Draft.h – line 38, hw7.cpp – line 86

*Testing*: After you have coded this method, menu option 1 should work properly. Test it and make sure!

## isValidPosition

*Parameters*: a string that contains the position (QB/WR/OL/TE/LB/CB/DL/S) that we want to check to see if there are any players in our array that match that position

*Returns*: boolean true or false if the position provided exists in our allPlayers[] array and if there are any available players that have not already been drafted

This method will iterate through the allPlayers[] array looking for at least one player position that matches the string that is passed in. If there is a match and that Player is available, this method will return true. If the user types in an invalid position (such as “quarterback”) or there are no remaining players in that position, the method will return false.

*Uncomment*: Draft.h – line 30, hw7.cpp – line 55

*Testing*: This method will be tested when you code the next method, but you will want to uncomment the lines listed in preparation for the next method! This method will also be used with pickBestByPosition() (method p) below. Make sure your code still compiles and runs as expected at this point.

## printAvailablePositionPlayers

*Parameters*: a string that contains the position on which we want to focus

*Returns*: nothing

Iterate through the array of allPlayers[] and print them to the screen if they are not currently taken in the draft.

*Uncomment*: Draft.h – line 39, hw7.cpp – lines 90 & 91

*Testing*: After you have coded this method, menu option 2 should work properly. Test with both valid and invalid input.

Now we’re ready to start drafting some players! Make sure that menu options 1 and 2 work as expected before continuing. Let’s work on our helper function first to make sure we’re able to add Players in just a moment.

## draftPlayer

*Parameters*: string, containing the drafting team’s name (pass-by-value)

int, the index at which the Player that’s being drafted is (pass-by-value)

*Returns*: nothing

This method will allow us to draft a player by adding the player at the given playerIndex to the given team. Once the Player is drafted, we need to set their status as taken, and set their team name to whomever drafted them. Now that a player has been drafted, the number of available players should be decremented. Then we want to inform the user who was added to which team (For example: “Player Viridian has been added to the Chiefs”).

*Uncomment*: Draft.h – line 17

*Testing*: This method will be tested with methods m, o, and p (shown below) – your code should still compile and run as usual at this point though.

Now that we can draft a Player, let’s work on those three methods, and any additional methods that need to be written for those to be successful.

## isValidName

*Parameters*: a string that contains the last name of a player (pass-by-value)

*Returns*: a boolean true or false if the name provided exists in our Player array

This method will iterate through the array of allPlayers[] checking to see if the last name provided is found in our array. If it is found, return true, otherwise return false. This method does *not* check if the player has already been drafted – that part is taken care of in the next method, pickByName (method m).

*Uncomment*: Draft.h – line 31

*Testing*: This method will be tested when the next method calls it

## pickByName

*Parameters*: string, the name of the team who is currently drafting (pass-by-value)

*Returns*: nothing

This method will check to make sure there are still players available to draft. If not, print a message to the user. If there are still players available, ask the user for the last name of the player they want to draft for their team. Loop while the name given is invalid. Once the name is validated, iterate through allPlayers[] array to find this Player’s index (where they appear in the array). When you get the index, make sure that the Player hasn’t already been drafted to another team. If they have, ask the user for another Player name and repeat.

*Uncomment*: Draft.h – line 32, hw7.cpp – line 95

*Testing*: After you’ve coded this method, menu option 3 should work correctly. Be sure to test trying to draft the same last name two times in a row – that will show that isValidName (method l above) is working as expected.

Okay! Now we are able to draft up to three Players for our team. We should notice that the number of players remaining at the end of the run has decreased by the number of players we selected. Be sure to test trying to add a player that doesn’t exist (such as “Clover”), and a player that’s already been added to your roster. It would also be a good idea to go ahead and get that last print method running. Let’s print all of the players that we’ve drafted on our team.

## printTeamPlayers

*Parameters*: a string that contains the team name

*Returns*: nothing

This method will iterate through the array of allPlayers[] and print them to the screen if they are a member of the team name that has been passed in.

*Uncomment*: Draft.h – line 40, hw7.cpp – line 159

*Testing*: Now when we run the code, and choose option 3 (the only one that allows us to draft so far), we should see a list of our chosen Players at the end of the output.

## pickBestOverall

*Parameters*: a string that contains the name of the drafting team (pass-by-value)

*Returns*: nothing

This method will choose the best remaining Player based on Overall Rank value. In this case, a smaller overall rank is better (i.e. an overall rank of 1 is best). We will iterate through all players in the allPlayers[] array looking for the lowest overall rank value. If the lowest overall rank value is not taken yet (so if they are taken, we can’t draft them to a second team), we will use their index value (where they are stored in the array, along with the given team name (passed in) to draft them.

*Uncomment*: Draft.h – line 33, hw7.cpp – line 100

*Testing*: Option 4 should now work. When you run the code at this point, and choose option 4 each time, your draft picks should be “Viridian”, “Lawrence”, and “Pitts”.

## pickBestByPosition

*Parameters*: a string that contains the position we’re searching for (pass-by-value)

a string that contains the name of the drafting team (pass-by-value)

*Returns*: nothing

This method will choose the best remaining Player based on their PositionRank value. Again, a smaller position rank is better (i.e. a position rank of 1 for QB is best). This will be implemented similarly to pickBestOverall above (method o), but with the additional constraint that we are looking for a specific position as well.

*Uncomment*: Draft.h – line 34, hw7.cpp – lines 105 & 106

*Testing*: Option 5 should now work. When you run the code at this point, and choose option 5 each time, and choose “QB” for the position each time, your draft picks should be “Viridian”, “Lawrence”, and “Lance”. If you test again, using only option 5, and choose “TE” for the position each time, your draft picks should be “Pitts” followed by a message that there are no other available players in that position. (This second one will test isValidPosition [method i] above).

Now that we’ve coded all of the needed methods in Draft.cpp, it’s time to test the ability of the other teams to pick players. The other three teams will simply pick the best overall player each time. Remove the block comment code (/\*) and (\*/) from lines 146 and 154 in hw7.cpp and run the program again. Does everything work as expected? When you input invalid data, does the program throw a segmentation fault (core dump) (many times this will be caused by trying to access an array element outside the bounds of your array!) or gracefully handle any errors and ask for more input?

# Implementation

The first step is to go through this document and the code to get an idea of how and why the methods are called. Code the methods in the order given, following the commentary provided, and test at every step. Suggested testing options are given for each method. Run your program and debug any errors that may occur.

You will only be coding in Draft.cpp. You should not edit Player.h or Player.cpp in any way. You will only be uncommenting lines of code in Draft.h. You will change a few things in hw7.cpp (adding your name, perhaps changing the teams in the draft), but mostly just uncommenting lines of code.

# Testing

Your program must run by using g++ on Turing. Test your program with a variety of inputs to check that it operates correctly for all of the requirements. You can create your code using your favorite editor or IDE.

Throughout your implementation of all of these methods, test your program to check that it operates correctly. Run your program with various input values and see that your error checking holds up. This is worth 10% of the homework grade so be sure that your program can handle these tests!

If you aren’t sure how to compile or execute your code, go the class website -> Lab Assignments -> [Online GDB Tutorial](http://uaf49268.ddns.uark.edu/~jgauch/csce2004_labs/labs/linux/tutorial0.pdf) for help.

## OnlineGDB

If you would like to design and implement your code on OnlineGDB first, you can create multiple files by clicking the upper left hand button on the screen and providing the name of the file. You will need to create five files in addition to your main.cpp file (Player.h, Player.cpp, Draft.h, Draft.cpp, and PlayerInfo.txt). When you run your code on OnlineGDB, pressing the green run button will compile all files at one time.

If you aren’t sure how to compile or execute your code, go the class website -> Lab Assignments -> [Online GDB Tutorial](http://uaf49268.ddns.uark.edu/~jgauch/csce2004_labs/labs/linux/tutorial0.pdf) for help.

You must be able to run your code on Turing, so you will need to transfer your files from your local computer to Turing. There are a few ways to accomplish this. First, log into Turing using:

ssh username@turing.csce.uark.edu

Choose one of the following options. There are more details available in the [technology guide](http://uaf49268.ddns.uark.edu/~jgauch/csce2004_labs/labs/linux/tutorial1.pdf) linked from the class website.

1. Copy and paste
2. Secure copy using the scp command (Linux and macOS only)
3. File transfer using software like FileZilla

Turing

If you would like to work directly on Turing, you will need to create and edit multiple files (you can use “vi”, “emacs”, or “nano” for editing). Make sure that all of your files reside in the same directory. You will need to specify every file name that you want to compile in the same g++ statement.

To compile:

g++ -Wall hw7.cpp Draft.cpp Player.cpp -o hw7.exe

To run:

./hw7.exe

## Typescript

You must submit a typescript of your project running on Turing. A typescript is a log file that contains a copy of all of your terminal text that demonstrates your program running. You will type “script” to start the typescript, and “exit” to finish. The following will generate a typescript of your program outputs, including a look into what is printed into your output file:

username@turing:~$ script homework7.txt

username@turing:~$ g++ -Wall hw7.cpp Draft.cpp Player.cpp -o hw7.exe

username@turing:~$ ./hw7.exe

username@turing:~$ ./hw7.exe *(<-- you will want to run your code multiple times for testing)*

*// \* Run program here with a variety of test cases \* //*

username@turing:~$ exit

You can transfer your files (.cpp, .h, and typescript) from Turing to your local computer using one of the techniques mentioned above.

# Documentation

When you have completed your C++ program, write a short report using the “Programming Project Report Template” describing what the objectives were, what you did, and the status of the program. Does your program work properly for all test cases? Are there any known problems? **If you are working with a partner, be sure to include their name on the report!** Save this project report in a separate document to be submitted electronically.

# Project Submission

In this class, we will be using electronic project submission to make sure that all students hand their programming projects and labs on time, and to perform automatic plagiarism analysis of all programs that are submitted. **IMPORTANT NOTE**: You must print your name and student ID at the very beginning of your program’s execution. If you have a partner, BOTH names and IDs must be printed.

When you have completed the tasks above go to Blackboard to upload your files. For full credit, you need to submit the following items:

1. **Three C++ (.cpp) files**: hw7.cpp, Draft.cpp, and Player.cpp

Please do not submit code in .docx, .txt, or any other format aside from .cpp!

1. **Two Header (.h) files**: Draft.h, Player.h
2. A **project report** in .docx or .pdf format following the template on the course website.
3. **One typescript** **file** that includes the compilation of your .cpp files, as well as running the program with multiple tests.

The dates on your electronic submission will be used to verify that you met the due date above. All late projects will receive reduced credit:

10% off if less than 1 day late,

20% off if less than 2 days late,

30% off if less than 3 days late,

no credit if more than 3 days late.

You will receive partial credit for all programs that compile even if they do not meet all program requirements, so handing projects in on time is highly recommended.

# Academic Honesty Statement

Students are expected to submit their own work on all programming projects, unless group projects have been explicitly assigned. Students are NOT allowed to distribute code to each other, or copy code from another individual or website. Students ARE allowed to use any materials on the class website, or in the textbook, or ask the instructor and/or GTAs for assistance.

This course will be using highly effective program comparison software to calculate the similarity of all programs to each other, and to homework assignments from previous semesters. Please do not be tempted to plagiarize from another student.

Violations of the policies above will be reported to the Provost's office and may result in a ZERO on the programming project, an F in the class, or suspension from the university, depending on the severity of the violation and any history of prior violations.

## Sample Output

Script started on 2021-04-17 21:59:32-0500

lstrothe@turing:~/FoundationsI $ g++ -Wall \*.cpp -o hw7.exe

lstrothe@turing:~/FoundationsI $ ./hw7.exe

This NFL Draft is written by Athena and Kaylee!

Welcome to the 2021 NFL Draft!

There are 25 players available. Here are all the players in the draft:

Name: Trevor Lawrence

Position: QB

Overall Rank: 2

Position Rank: 2

// Many more players printed to the screen – cut for length //

Name: Christian Barmore

Position: DL

Overall Rank: 25

Position Rank: 3

=============== ROUND 1 ===============

-----Team: Chiefs -----

Please Choose From the Following Options

1. Print all available players

2. Print available players based on position

3. Pick by player name

4. Pick best overall ranked player

5. Pick best ranked player based on position

6. Skip draft pick

Enter your choice >> 3

Please input the name of the player you wish to draft >> Jones

[+] Player Jones has been drafted.

[+] Player Viridian has been drafted.

[+] Player Lawrence has been drafted.

[+] Player Pitts has been drafted.

=============== ROUND 2 ===============

-----Team: Chiefs -----

Please Choose From the Following Options

1. Print all available players

2. Print available players based on position

3. Pick by player name

4. Pick best overall ranked player

5. Pick best ranked player based on position

6. Skip draft pick

Enter your choice >> J 3

Please input the name of the player you wish to draft >> Jones

That player has already been drafted. Please try again.

Please input the name of the player you wish to draft >> Smith

[+] Player Smith has been drafted.

[+] Player Toney has been drafted.

[+] Player Darrisaw has been drafted.

[+] Player Farley has been drafted.

=============== ROUND 3 ===============

-----Team: Chiefs -----

Please Choose From the Following Options

1. Print all available players

2. Print available players based on position

3. Pick by player name

4. Pick best overall ranked player

5. Pick best ranked player based on position

6. Skip draft pick

Enter your choice >> 4

[+] Player Lance has been drafted.

[+] Player Vera-Tucker has been drafted.

[+] Player Parsons has been drafted.

[+] Player Wilson has been drafted.

DRAFT PICKING FOR 2021 IS OVER. HERE ARE THE PLAYERS ADDED TO YOUR TEAM:

Name: DeVonta Smith

Position: WR

Overall Rank: 9

Position Rank: 2

Name: Trey Lance

Position: QB

Overall Rank: 7

Position Rank: 3

Name: Mac Jones

Position: QB

Overall Rank: 22

Position Rank: 5

After all teams have made their choices, there are 13 players remaining.

Congratulations Jessie and James for yet another successful venture, this time in NFL drafting!

Giovanni would be proud :-)

lstrothe@turing:~/FoundationsI $ ./hw7.exe

This NFL Draft is written by Athena and Kaylee!

Welcome to the 2021 NFL Draft!

There are 25 players available. Here are all the players in the draft:

// Players printed to the screen – cut for length //

=============== ROUND 1 ===============

-----Team: Chiefs -----

Please Choose From the Following Options

1. Print all available players

2. Print available players based on position

3. Pick by player name

4. Pick best overall ranked player

5. Pick best ranked player based on position

6. Skip draft pick

Enter your choice >> 5

Enter the position are you interested in >> TE

[+] Player Pitts has been drafted.

[+] Player Viridian has been drafted.

[+] Player Lawrence has been drafted.

[+] Player Toney has been drafted.

=============== ROUND 2 ===============

-----Team: Chiefs -----

Please Choose From the Following Options

1. Print all available players

2. Print available players based on position

3. Pick by player name

4. Pick best overall ranked player

5. Pick best ranked player based on position

6. Skip draft pick

Enter your choice >> 5

Enter the position are you interested in >> TE

Sorry, there are no available players for that position.

Acceptable positions (although not guaranteed to be available) are: QB, WR, OL, TE, LB, CB, DL, S

Enter the position are you interested in: qb

Sorry, there are no available players for that position.

Acceptable positions (although not guaranteed to be available) are: QB, WR, OL, TE, LB, CB, DL, S

Enter the position are you interested in: QB

[+] Player Lance has been drafted.

[+] Player Darrisaw has been drafted.

[+] Player Farley has been drafted.

[+] Player Vera-Tucker has been drafted.

=============== ROUND 3 ===============

-----Team: Chiefs -----

Please Choose From the Following Options

1. Print all available players

2. Print available players based on position

3. Pick by player name

4. Pick best overall ranked player

5. Pick best ranked player based on position

6. Skip draft pick

Enter your choice >> 6

[+] Player Smith has been drafted.

[+] Player Parsons has been drafted.

[+] Player Wilson has been drafted.

DRAFT PICKING FOR 2021 IS OVER. HERE ARE THE PLAYERS ADDED TO YOUR TEAM:

Name: Kyle Pitts

Position: TE

Overall Rank: 3

Position Rank: 1

Name: Trey Lance

Position: QB

Overall Rank: 7

Position Rank: 3

After all teams have made their choices, there are 14 players remaining.

Congratulations Jessie and James for yet another successful venture, this time in NFL drafting!

Giovanni would be proud :-)

lstrothe@turing:~/FoundationsI $ exit

exit

Script done on 2021-04-17 22:00:59-0500