# CSCE 221 Cover Page
# Homework #1
# Due September 18 at midnight to CSNet

First Name Clayton Last Name Dittman UIN 824001276

User Name Claywd /claywd 1210 E-mail address Clayton.dittman@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero. According to the University Regulations, Section 42, scholastic dishonesty are including: acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used. Disciplinary actions range from grade penalties to expulsion read more: Aggie Honor System Office

| Type of sources | Cplusplus.com | Stackexchange.com | |
|---|---|---|---|
| People | | | |
| Web pages (provide URL) | | | |
| Printed material | | | |
| Other Sources | | | |

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.

"On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work."

Your Name Date

Problem 1:

A) This code is separated into four parts. Main which just initializes the functions, game.h which holds the game declarations, game.cpp defines those definitions, and player.h which includes the base class player and the definitions for it's two derived classes. The compile command "g++ -std=c++11 *.cpp" will compile the code correctly for you and this code was written on the build.tamu.edu server using c++11 standards.
    a. The program will allow you to test whatever ranges you like or you can let the computer automatically pick from the set of ranges in this assignment.

B) After choosing if your role in the game the program will allow you to choose to set the range manually or get the range automatically at then proceed to prompt you for your guess or show you the computers guess. Ideally, these results would be written out to a file and stored. If I had more time I would have finished writing the section of code which imported data from a text file and used it in the program.

| **Range [1..n]** | **True Answer** | **# guesses/comparison** | **Results of Formula in 'c'** |
|---|---|---|---|
| [1,1] | 1 | 1 | 1 |
| [1,2] | 2 | 1 | 1 |
| [1,4] | 4 | 1 | 1 |
| [1,8] | 8 | 1 | 1 |
| [1,16] | 16 | 1 | 1 |
| [1,32] | 32 | 1 | 1 |
| [1,64] | 64 | 1 | 1 |
| [1,128] | 128 | 1 | 1 |
| [1,256] | 256 | 1 | 1 |
| [1,512] | 512 | 1 | 1 |
| [1,1024] | 1024 | 1 | 1 |
| [1,2048] | 2048 | 1 | 1 |

C) This algorithm is classified as $O(n^2)$.

Problem 2:

```
/*pseudocode
void remove_at_rank(reference to a vector, int k)
{
        if element is at the back of vector, {remove it}
        else { element_at_rank[k] = element_at_end; remove last element in vector}
}
*/
```

```
/*pseudocode
void remove_at_rank(reference to a vector, int k)
{
    if element is at the back of vector, {remove it}
    else { element_at_rank[k] = element_at_end; remove last element in vector}
}
*/
```

Here we basically just overwrite the data member to be replaced with the last element in the object and then delete the original object we copied. This allows for a time efficiency of O(1).

Problem 3:

```
int main()
{
        string input;

        cout << "Please enter a string: ";
        cin >> input;

        if (input == string(input.rbegin(), input.rend()))
        {
                cout << input << " is a palindrome";
        }
return 0;
}
```

```
int main()
{
    string input;

    cout << "Please enter a string: ";
    cin >> input;

    if (input == string(input.rbegin(), input.rend()))
    {
        cout << input << " is a palindrome";
    }
return 0;
}
```

Problem 4:

This is possible when there is an upper bound on the algorithm they are testing with where the upper bound is 100. As n gets large a function like binary search would resemble Al's $O(n^2)$ time efficiency while a quicksort would resemble Bob's $O(nlog(n)$ time efficiency.

Problem 5:

A) Ex1= $O(n)$
B) Ex2= $O(n^2 + n)$
C) Ex2= $O(n^2)$