

Stackexchange.com

Cplusplus.com

Claywd/claywd1210

Clayton.dittman@tamu.edu

824001276

Dittman

Clayton

Problem 1:

1. 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.
   1. 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.
2. 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 Anser n** | **# 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 |

1. This algorithm is classified as

Problem 2:

/\*pseudocode

void remove\_at\_rank(reference to a vector, int k)

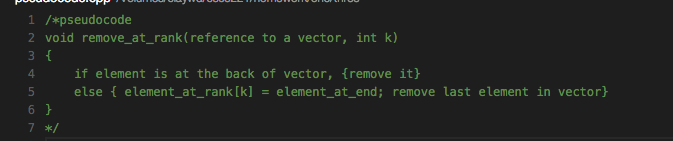
{

if element is at the back of vector, {remove it}

else { element\_at\_rank[k] = element\_at\_end; remove last element in vector}

}

\*/



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;

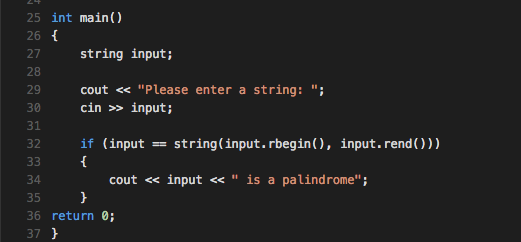
if (input == string(input.rbegin(), input.rend()))

{

cout << input << " is a palindrome";

}

return 0;

}

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 time efficiency while a quicksort would resemble Bob’s time efficiency.

Problem 5:

1. Ex1=
2. Ex2=
3. Ex2=