# Programming Assignment 2

## Most Valuable Player Redux

**(More than one way to skin a Wildcat)**

Due Tuesday 14 February 2012 (before midnight)

**Overview**
The learning objective of this project is for you to understand how variable memory can be dynamically allocated and the use of compound data structures. In this assignment you will enhance your program that was written for assignment 1 to use a struct to hold the statistics for an individual player.

**Description**
As with program 1, this program will read statistical data for a set of basketball players, calculate shooting percentages and points per 48 minutes, find the top offensive and defensive players and crown an MVP. The results will again be written to the console as they were in your first programming assignment.

**Specifications**
In program 1 your program used individual variables to hold the various statistics for each player. In this program you will define a ***compound structure*** that will combine all of these datum under a single variable (structure) name. The new structure will have a type specifier of **Player**. For each set of player data you will dynamically allocate a new Player structure and then populate the individual elements (Text Ch 7.7).

With program 1, parallel vectors were used to maintain collections of players. In program 2 you are to use a single vector, the elements of which will be pointers to individual player *structures* allocated to store player information. This change in structure will give you an alternative view of how to manipulate data in C++.

You were provided with four input functions for use in program 1. In program 2, you will be writing your own input function. You can examine the input functions from program 1 and are free to re-use or re-write any of the code from those functions in your new function. This function will be named:

```
bool getPlayerStats(Player*)
```

You will be responsible for implementing the following functions:

**bool getPlayerStats(Player*)**

input: pointer to Player

output: true – if data was found in file
false – if end of file condition occurred

side effect: Player structure passed as input parameter (pointer) will be populated with data from the input file.

**void calcPercentages(Player*)**

input: pointer to Player

output: void

side effect: FG PCT, 3PT PCT, FT PCT, Total Rebounds, and Points Per 48 will be calculated and populated in the Player structure.

**void findTopOffensivePlayer(vector<Player*>)**

input: vector of pointers to Player

output: void

side effect: Offensive Rating for each Player will be calculated and populated in Player structures. Function will display (to console) name of top offensive player.

**void findTopDefensivePlayer(vector<Player*>)**

input: vector of pointers to Player

output: void

side effect: Defensive Rating for each Player will be calculated and populated in Player structures. Function will display (to console) name of top defensive player.

**void findMostValuablePlayer(vector<Player*>)**

input: vector of pointers to Player

output: void

side effect: Function will display (to console) name of most valuable player.

**void outputPlayers(vector<Player*>)**

input: vector of pointers to Player

output: void

side effect: Function will report statistics for all players including totals.

**void deallocatePointers(vector<Player*>)**

input: vector of pointers to Player

output: void

side effect: Function will deallocate all heap memory allocated for Player structures.

**Notes**
All functions must have the signatures established by the specifications. No global variables are allowed outside of the ifstream variable for the input file.