# Coms 262 The `Player` Class Spring 2017

## Data Members

The `Player` class has three data members.

- `string m_name` – The player's name.
- `int m_size` – The number of bids (the same as the number of assets).
- `double* m_bids` – A pointer to a dynamically allocated array of bids (`double`s).

## Member Functions

### Constructors

- `Player();`

  Constructs a default `Player` object with name `"no name"`, size `0`, and the pointer set to `NULL`.

- `Player(const Player& p);`

  Constructs a `Player` object as a copy of the specified `Player`.

- `Player(string nm, int sz);`

  Constructs a `Player` object with the specified values for `m_name` and `m_size`. If `m_size` is `0`, then `m_bids` will be set to `NULL`. If `m_size` is positive, then an array of `double`s of that size will be allocated dynamically and `m_bids` will point to that array. The array elements should be initialized to `0`.

### The Destructor

- `~Player();`

  Destroys a `Player` object. Deallocates the memory allocated to the object for the array of bids.

### Inspectors

- `string name() const;`

  Returns the name of the player.

- `int size() const;`

  Returns the number of bids.

- `double bid(int i) const;`

  Returns the `i`$^{\text{th}}$ bid as an *r*-value.

- `double& bid(int i);`

  Returns the `i`$^{\text{th}}$ bid as an *l*-value.

### Mutators

- `void name(const string& nm);`

  Sets the name of the player to the specified string.

- `void size(int sz);`

  Sets the number of bids to the specified value. Then it deallocates the current array of bids. If `sz` is positive, an array of `double`s of the new size will be allocated dynamically and `m_bids` will point to that array. Otherwise, `m_bids` should be set to `NULL`.

- `void bid(int i, double b);`

  Sets the `i`th bid to the specified value `b`.

### Facilitators

- `void input(istream& in);`

  Reads a `Player` object. The format of a `Player` object is the player's name (no embedded white space), followed by the number of bids, followed by the bids. The values are separated by blanks.

- `void output(ostream& out) const;`

  Writes a `Player` object to the output stream. The output format is the same as the input format.

### Operators

- `Player& operator=(const Player& p);`

  Assigns the value of the specified `Player` object to the invoking `Player`.

- `double operator[](int i) const;`

  Returns the `i`th bid as an *r*-value.

- `double& operator[](int i);`

  Returns the `i`th bid as an *l*-value.

### Other Member Functions

- `void makeEmpty();`

  Sets the invoking `Player` to the default state.

- `void makeCopy(const Player& p) const;`

  Makes the invoking `Player` a copy of the specified `Player`.

- `double totalValue() const;`

  Returns the total value of the player's bids.

## Non-member Operators

- `istream& operator>>(istream& in, Player& p)`

  Reads a `Player` object from the input stream.

- `ostream& operator<<(ostream& out, const Player& p)`

  Writes a `Player` object to the output stream.