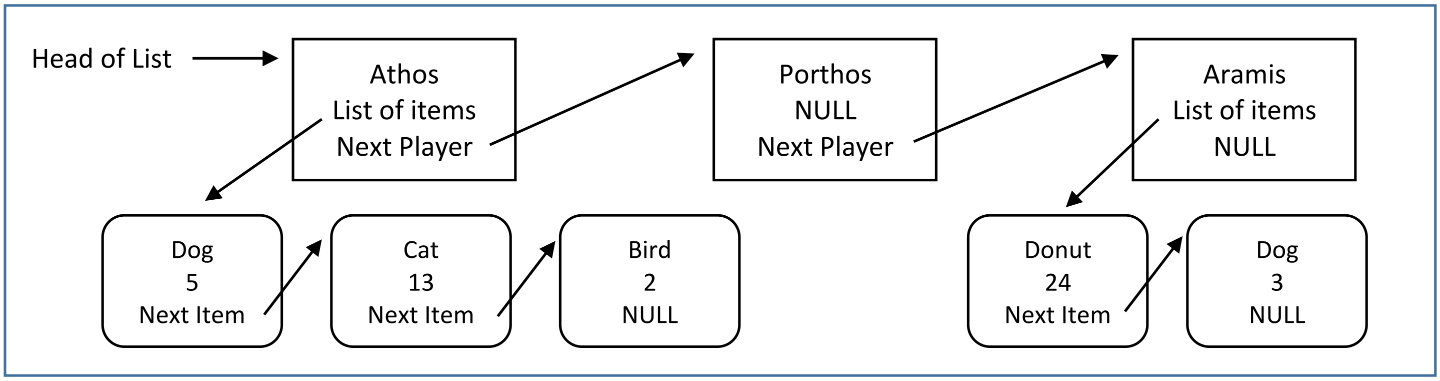
CS 100 Project Four – Spring 2016

Project Overview: Build and manage a linked list that contains the players in a game and the items that each player possesses.



The picture above captures a situation where you have three players (Athos, Porthos and Aramis). Athos has 5 dog, 13 cat and 2 birds. Porthos has nothing in his possession. Aramis has 24 donut and 3 dog.

Getting Started

- This program is written in pieces. It has a main routine (`main.c`) that reads commands from a file `data`. The functions that actually carry out the work are found in `game.c`. Several of these files already exist. You can copy the files `main.c` and `game.h` and `game.c` and a sample data file (`data`) from **troll.cs.ua.edu/cs100/projects/project4**.
- After you copy the four files listed above to your directory, compile the program (**gcc –Wall main.c game.c scanner.c**) and run it via **./a.out data**. Look at this output and the file **data** to make sure you understand what the program is doing.

What You Need To Do

1. Create a directory called `project4` on your machine. In that directory, you will have files named `game.c` and `game.h` and `main.c` and `Makefile`, as well as `scanner.c` and `scanner.h` and a sample `data` file.
2. **You should NOT modify the files `main.c` or `game.h` at all, all your edits should be to `game.c` only.**
3. The specific functions that you must write, all of which are found in `game.c`, are shown below:

Function	Points	Signature	Comments
add	10	`Players *add(char *);`	Adds a new player at start of list
addItem	15	`void addItem(Players *, char *p, char *i, int n);`	Adds **n** items (item **i**) to player **p**, who might already have some of that item
countPlayers	5	`int countPlayers(Players *);`	Returns the total number of players
countItems	10	`int countItems(Players *,char *p);`	Returns the total items for player **p**
numItems	10	`int numItems(Players *, char *i);`	Returns how many instances of item **i** exist (sum of that item for all players)
orderedAdd & orderedAddItem	20	An **add** routine (and **addItem** routine) that adds players and items in alphabetical order	
gives	15	`void gives(Players *, char *p1, char *p2, char *i);`	Player **p1** gives all of his/her item **i** to player **p2**

4. In addition to the points listed above, you get **5** points for submitting a **Makefile** and **5** points for a properly commented `game.c` routine that compiles successfully and **5** points for submitting your own sample `data` file.
5. We recommend that you build this project in pieces.
   - First, get **add** and **addItem** working
   - Second, get **countPlayers** and **countItems** and **itemCount** working
   - Third, write **orderedAdd** and **orderedAddItem** so that items are added in alphabetical order
   - Finally, get the **gives** function working (which involves removing from a list)
6. You can assume that all input for this program will be legal. You don't have to worry about invalid input/operations.

When you are ready to submit your project:

- Make sure your program runs properly on **cs-intro.ua.edu**. Your program is graded on that system.
- Bundle your `project4` directory (be sure to include **game.c** and a **Makefile**) into a single (compressed) zip file.
- Once you have a compressed zip file that contains your `project4`, submit that file to Blackboard.

Project Four is due at 5:00pm on Friday, April 8. Late projects are not accepted.