California State University, Sacramento
College of Engineering and Computer Science

Computer Science 35: Introduction to Computer Architecture

Fall 2022 – Lab 5 – *Sorting Hat*

Overview

Every student at the famous ***Hogwarts School of Witchcraft and Wizardry*** is sorted into one of the Four Houses with each having its own common room and attached dormitories.

Your nerves are quite calm – well, that is, somewhat calm after drinking a cup of delicious, magical, hot cocoa.

Ninji of the Nice, an eloquently carved statue of a cat, stretches and yawns. "It's time now. Have a nice sorting young human kitten". The statue then lays down on it pedestal, crosses its little stone arms, and promptly falls asleep.

You are bustled into the Great Hall and huddle with your fellow students as the eyes of the older students fall upon you. With a lurch in your stomach, you realize that the hot chocolate has evidently worn off. Or, perhaps, you are so nervous, that the magical hot cocoa isn't strong enough. But, what if you didn't get the hot cocoa in the first place? You might have passed out? Are you going to pass out? Oh, my goodness! You are going to going to pass out!

Suddenly, you are jerked back to reality and your inner argument is extinguished. A rip along the base of the Sorting Hat had opened, and it had begun to sing.

> ***You might belong in Gryffindor… where dwell the brave at heart.***
> ***Their daring, nerve, and chivalry… set Gryffindors apart.***
>
> ***Or yet in wise old Ravenclaw… if you've a ready mind.***
> ***Where those of wit and learning… will always find their kind.***
>
> ***You might belong in Hufflepuff…where they are just and loyal.***
> ***Those patient Hufflepuffs are true… and unafraid of toil.***
>
> ***Or perhaps in Slytherin… you'll make your real friends.***
> ***Those cunning folks use any means… to achieve their ends.***

When the Sorting Hat finishes its song, the Great Hall bursts into applause. Professor McGonagall raises her hand for silence, and it falls upon the room almost immediately. With a nod, she reaches into her cloak and removes a long, winding, piece of parchment, pushes her square-rimmed glasses down her nose, and begins to read.

One by one, the students are called by name. They run forward and the Sorting Hat is placed gently upon their heads. Sometimes, the Sorting Hat immediately yells out their house names, but for some students, it takes several minutes. Once the house is announced, however, the student joins their new house tables accompanied with thunderous applause.

Then, your name is called. With great trepidation, you approach the stool. Moments later, the hat drops onto your head and falls past your eyes. All you see is darkness.

A voice, not of your own, echoes in your head. "Ah, let's have a look what's in here. Hmmm. I see intelligence. Yes, yes. You have a good mind. You will do well in Ravenclaw."

You swell with pride at these words. "Yes, I am a muggle programmer".

The voice returns, sounding contemplative and suspicious. "Ah, a programmer, you say? I know of these things. Sneaky… clever. You will also do well in Slytherin."

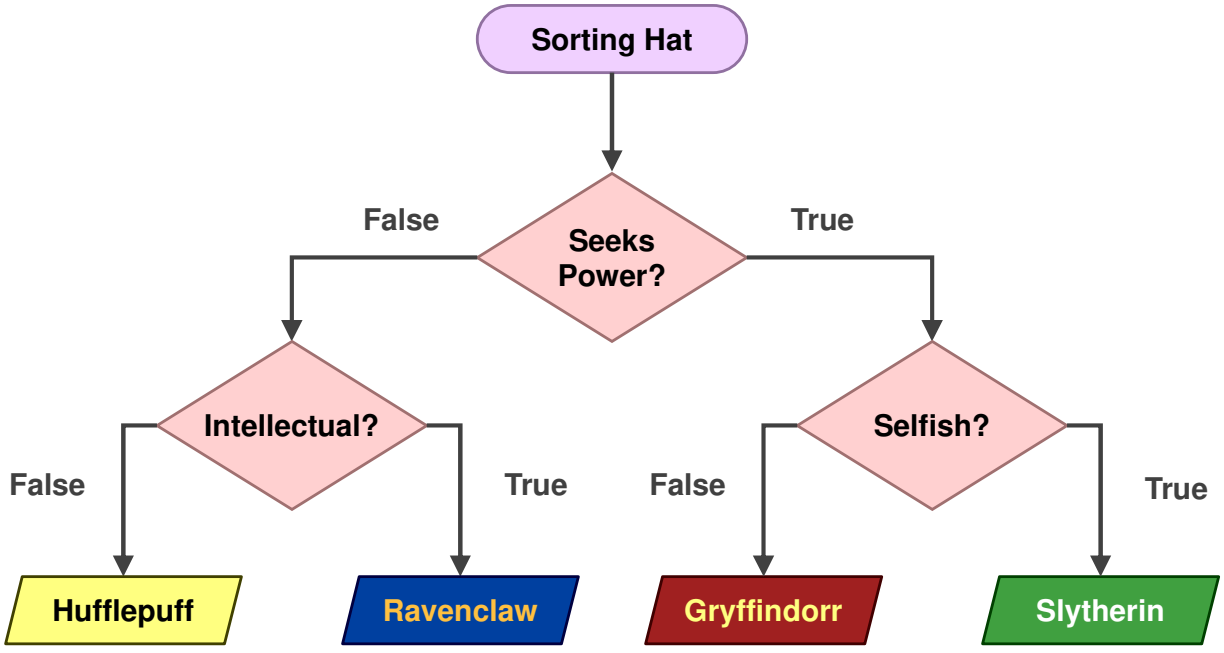
"Hmmm.", the voice continues, "I see bravery here too. You are taking a class from a known lunatic… You will do well in Gryffindor. But, ah, again, you are a programmer… a hard worker… perhaps Hufflepuff then."

The voice sighs. "This may take a while…"

The Sorting Hat's Algorithm

While the Sorting Hat seems mysterious and powerful, it follows a basic nested If Statement. Well, this isn't really true, but it's your instructor's attempt to turn it into a programming assignment.

So, for this assignment, let's **assume** that it simply senses the answers to a few questions and then puts the student into the correct house (hopefully). The logic is as follows:



Your Task

So, for your assignment, you get to create a program to recreate the Sorting Hat. You will ask questions and the user will respond with a numeric value. Based on their answers, you will print which house they will be put into.

Have Fun!

You don't have to use the four houses from Harry Potter. Create your own four categories. Naturally, the questions will be different, but the overall approach is the same. The following are some example categories.

- What class in a role player game are you?
- What should you have for dinner?
- What meme are you most resemble?
- Which SpongeBob character are you?
- Which Rick and Morty alien are you?
- What kind pet should you get – species, breed, etc...
- What political philosophy are you?
- What music you should listen to?
- etc...

Examples

This is the output from one possible solution. Your solution doesn't have to look exactly like the example below. But, make sure to fulfill all the requirements. Input is displayed in **blue**.

```
Welcome to Hogwarts!
The Sorting Hat is being placed upon your head.


Do you wish to command/control others (y/n)?
n
Do you find contentment in reading (y/n)?
y
Ravenclaw!
```

```
Welcome to Hogwarts!
The Sorting Hat is being placed upon your head.

Do you wish to command/control others (y/n)?
y
Is it better to help yourself more than others (y/n)?
y
Slytherin!
```

```
Welcome to Hogwarts!
The Sorting Hat is being placed upon your head.

Do you wish to command/control others (y/n)?
y
Is it better to help yourself more than others (y/n)?
n
Gryffindor!
```

Tips

Reading Characters

CSC 35 Library has a subroutine called "ScanChar" that will read a byte from the keyboard and store it into **dl**.

```
call ScanChar              #dl = input;
```

How to approach the problem

- When a character is stored in **dl**, it will contain an ASCII character code. You will need to use these in your **cmp** instructions. Look-up an ASCII chart online.
- Like all labs, **build it in pieces**. First get a single If-Statement to work. Then, you can work on to more detailed ones.
- All labels **must** be unique. Choose your names well.
- Assembly doesn't have blocks… so don't think in those terms. Your program will be structured far differently from programs in Java.

Requirements

Now work on each of the requirements below one at a time. You will turn in the final program, but incremental design is best for labs. You must think of a solution on your own.

The requirements are as follows:

1. Put your name and section # in a comment at the very top of your program.
2. Display text explaining that theme of your program. You don't have create the Hogwarts one that I used. Use your imagination.
3. Input each choice (with prompts). You **must** input a character. Any lab that uses ScanInt will receive a zero.
4. Implement the logic in the flowchart above. Use nested ifs
5. Display the output for all four possibilities.

Submitting Your Lab

> ⚠ **This activity may only be submitted in Intel Format.**
>
> **Using AT&T format will result in a zero. Any work from a prior semester will receive a zero.**

To submit your lab, you must run Alpine by typing the following and, then, enter your username and password.

```
alpine
```

To submit your lab, send the assembly file (do not send the a.out or the object file to:

```
dcook@csus.edu
```

UNIX Commands

Editing

Action	Command	Notes
Edit File	**nano** *filename*	"Nano" is an easy to use text editor.
E-Mail	**alpine**	"Alpine" is text-based e-mail application. You will e-mail your assignments it.
Assemble File	**as -o** *object source*	Don't mix up the *objectfile* and *asmfile* fields. It will destroy your program!
Link File	**ld -o** *exe object(s)*	Link and create an executable file from one (or more) object files

Folder Navigation

Action	Command	Description
Change current folder	**cd** *foldername*	"Changes Directory"
Go to parent folder	**cd ..**	Think of it as the "back button".
Show current folder	**pwd**	Gives the current a file path
List files	**ls**	Lists the files in current directory.

File Organization

Action	Command	Description
Create folder	**mkdir** *foldername*	Folders are called directories in UNIX.
Copy file	**cp** *oldfile newfile*	Make a copy of an existing file
Move file	**mv** *filename foldername*	Moves a file to a destination folder
Rename file	**mv** *oldname newname*	Note: same command as "move".
Delete file	**rm** *filename*	Remove (delete) a file. There is **no** undo.