601.220 Intermediate Programming

Spring 2023, Day 4 (Jan 30th)

Today's agenda

- Review exercises 3-a and 3-b
- C logical operators and control flow
- Exercise 4

Reminders

- HW0 is due this Friday (Feb 3rd)

Exercise 3-a review

- Goal: clone and use your private repo
  - Use `git config` to set the name and email address you want to appear in log messages
  - Use the `git clone` command
- Note that the instructions said to use the "https" version of the Git URL, but it's better to use the "ssh" version of the Git URL
  - Requires that you have created an ssh key and added it to your Github account
  - See us if you need help getting this to work

Exercise 3-a review

Cloning your private repo (assuming you're using ssh):

`git clone git@github.com:jhu-ip/2023-spring-student-JHEDID.`
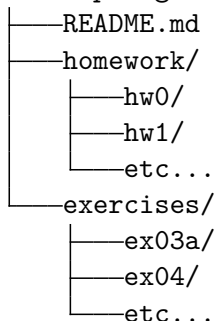
(where `JHEDID` is your JHED ID.)

Creates a directory "2023-spring-student-JHEDID", which is the local clone of your private repository.

Exercise 3-a review

Suggested organization for your private repository:

```
2023-spring-student-JHEDID/
├──README.md
├──homework/
│    ├──hw0/
│    ├──hw1/
│    └──etc...
└──exercises/
     ├──ex03a/
     ├──ex04/
     └──etc...
```

Exercise 3-a review

Working on an exercise or homework:

- Create a directory for it within your private repository (e.g., `mkdir -p exercises/ex03a`)
- Create and edit files
- Run and test program
- Use `git add` and `git commit` to commit your work
- Use `git push` to push commits to your remote Git repository (on Github)

Exercise 3-a review

Example of creating a zipfile for an exercise or homework.

On ugrad:

```
cd ~
cd 2023-spring-student-JHEDID/exercises/ex03a
git log > gitlog.txt
zip ex03.zip README gitlog.txt
cp ex03.zip ~
```

On your local machine:

```
scp USERNAME@ugradx.cs.jhu.edu:ex03.zip .
unzip -l ex03.zip
```

Exercise 3-a review

Note about running ssh and scp on Windows:

You can use PuTTY and pscp. However, you can also run
PowerShell, and then just run "ssh" and "scp" commands from
there. This works more or less exactly the same way as if you were
using a Mac or Linux system.

To run PowerShell, just type "PowerShell" in the search box.

Exercise 3-b review

- Goal: clone the "public" repository for the course, copy starter files for exercise
- As with cloning your private repo, it's best to use the "ssh" Git URL

Exercise 3-b review

Cloning the "public" repository

`git clone git@github.com:jhu-ip/cs220-sp23-public.git`

Note that you cannot push changes to the public repository, and it's not intended that you will modify any of the files in your clone of the public repository.

Working on exercise

```
cd ~
mkdir temp
cd temp
cp ~/cs220-sp23-public/exercises/ex03b/hello_world.c .
```

Note that in this exercise we had you copy the starter code into a "temp" directory. In the future, you will create a directory in your private repository for each exercise, and copy the starter code there.

Day 4 recap questions

1. Which one is the logical "and" operator in C, && or & or both?
2. Which one is the logical "negation" operator in C, ~ or ! or both?
3. What is the result of evaluating `(34 + 2) / 40 || 80 > 'A' && 15 % 4` ?
4. What does the keyword **break** do in a control structure?
5. What does the keyword **continue** do in loops?
6. How many times is the *initialize* statement in a *for loop* executed?

1. Which one is the logical "and" operator in C, && or & or both?

&& is the logical "and" operator.

& is the *bitwise* "and" operator.

If you're expressing a condition (for an if, if/else, or loop) then you want &&.

2. Which one is the logical "negation" operator in C, ~ or ! or both?

! is the logical negation operator.

~ is bitwise complement, a.k.a. bitwise negation.

3. What is the result of evaluating (34 + 2) / 40 ||
80 > 'A' && 15 % 4 ?

Key ideas:

- 0 is false, and all other integer values are true
- all logical operators are guaranteed to produce either 0 or 1 as
  a result

4. What does the keyword **break** do in a control structure?

`break` jumps to the code that follows the control structure.

So, it can be used to terminate a loop early.

Important: `break` (and `continue`) can make loops harder to understand. They should be used very sparingly, and only when they truly simplify the code.

break and switch statements

break is also used to end a case (or group of cases) in a switch statement. This is a *necessary* use of break (as opposed to break in the body of a loop, which is never strictly necessary.)

5. What does the keyword **continue** do in loops?

continue jumps to the top of the loop.

Again, this can make the loop harder to understand, so use sparingly.

6. How many times is the *initialize* statement in a *for loop* executed?

Once, before the loop starts.

Assume i is an int variable. These loops are equivalent:

```c
// while loop
i = 1;
while (i <= 10) {
  printf("%d\n", i);
  i++;
}

// for loop
for (i = 1; i <= 10; i++) {
  printf("%d\n", i);
}
```

Exercise 4

GPA calculator program: practice

- starting an exercise using code in the public repository
- input using `scanf`
- output using `printf`
- loops
- `switch` statements

Notes

Notes

Notes

Notes

Notes