New target: `printrun`

- New build target in Makefile: `printrun`
- A log of GeekOS I/O is output to `build/out.txt`
- Example usage:
  ```bash```
  ```
  ~/project5/build$ make printrun
  ```
Debugging locally

- Open two terminal windows
- Terminal 1:
  
  ```bash
  ~/project5/build$ make dbgrun
  ```
- Terminal 2:
  
  ```bash
  ~/project5/build$ make dbg
  ```
- Terminal 2 is running gdb, and the other is running GeekOS in QEMU
- In gdb, type `continue` to begin
Debugging remotely (on Linuxlab)

- Open two terminal windows
- Terminal 1:
  ```
  ~$ ssh -Y <username>@linuxlab.csic.umd.edu
  [<username>@<compname> ~]$ cd project5/build
  [<username>@<compname> build]$ make dbgrun
  ```
- Terminal 2: (note the `<compname>` from Terminal 1)
  ```
  ~$ ssh -Y <username>@<compname>.csic.umd.edu
  [<username>@<compname> ~]$ cd project5/build
  [<username>@<compname> build]$ make dbgrun
  ```
- Terminal 2 is running gdb, and the other is running GeekOS in QEMU
- In gdb, type `continue` to begin
Debugging example: ROT13

New system call `ROT13(char *str)`:

```c
// state->ebx points to the string, state->ecx = string length
static int Sys_ROT13(struct Interrupt_State *state) {
    int i, n = state->ecx;
    char *str = 0;
    if (Copy_User_String(state->ebx, n, 1023, &str) != 0) return -1;
    for (i = 0; i < n; i++) {
        if (str[i] >= 'A' && str[i] <= 'Z')
        else if (str[i] >= 'a' && str[i] <= 'z')
            Print("%c", str[i] + ((str[i] + 13 <= 'z') ? 13 : +13));
        else Print("%c", str[i]);
    }
    Free(str);
    return 0;
}
```

New user program `src/user/rot13.c`:

```c
int main(int argc, char **argv) {
    int i;
    if (argc == 1) Print("Usage: rot13 [STRING] ...
"));
    else
        for (i = 1; i < argc; i++) {
            ROT13(argv[i]);
            (i+1 < argc) ? Print(" ") : Print("\n");
        }
    return 0;
}
```
Problem: incorrect output for some input to rot13

Welcome to GeekOS!
Spawning init process (/c/shell.exe)
$ rot13 ONYX cat
BALK pn_
$ rot13 png
}\{t
$ exit
DONE!
Init process exited with code 0
Debug ROT13

- Start debugging: Terminal 1 (QEMU/GeekOS) and Terminal 2 (gdb)
- In Terminal 2 gdb, type `break Sys_ROT13` to set a breakpoint at `Sys_ROT13`, and then type `continue` to begin
- In GeekOS, run `rot13 ONYX cat`
- In gdb, it should stop at the beginning of `Sys_ROT13`.
  - `next`: go to the next instruction
  - `step`: go to the next instruction (or step into a function call)
  - `continue`: go to the next breakpoint
  - `print <var>/<ex>/<eq>` to print a variable, expression, or equation. E.g. `print (char) (str[1]+13 <= 'Z')`
- On the second call to `Sys_ROT13`, note that the ’t’ in ’cat’ should evaluate to false, so we should print `str[i] - 13`, but we print `str[i] + 13` instead.
Debug ROT13

- On the second call to `Sys_ROT13`, note that the 't' in 'cat' should evaluate to false for the statement
  
  \[(str[i] + 13 <= \text{'z'})\]

- Thus, logically we know to print \(str[i] - 13\), but instead we print \(str[i] + 13\).

- This shows the bug: the `else` branch in the ternary operator should have \(-13\) instead of \(+13\).

- Check more on gdb online if you’re unfamiliar.