Code Examples from Lectures on filters

COMP2041|COMP9044 19T2

cat.c

Simple /bin/cat emulation.

```
#include <stdio.h>
#include <stdlib.h>

// write bytes of stream to stdout
void process_stream(FILE *in) {
    while (1) {
        int ch = fgetc(in);
        if (ch == EOF)
             break;
        if (fputc(ch, stdout) == EOF) {
            fprintf(stderr, "cat:");
            perror("");
            exit(1);
        }
    }
}

// process files given as arguments
// if no arguments process stdin
int main(int argc, char *argv[]) {
    if (argc == 1)
        process_stream(stdin);
    else
        for (int i = 1; i < argc; i++) {
            FILE *in = fopen(argv[i], "r");
            if (in == NULL) {
                fprintf(stderr, "%s: %s: ", argv[0], argv[i]);
                perror("");
                return 1;
            }
            process_stream(in);
            fclose(in);
        }
    return 0;
}
```

wc.c

Simple /usr/bin/wc emulation.

```c
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>

// count lines, words, chars in stream
void process_stream(FILE *in) {
    int n_lines = 0, n_words = 0, n_chars = 0;
    int in_word = 0, c;
    while ((c = fgetc(in)) != EOF) {
        n_chars++;
        if (c == '\n')
            n_lines++;
        if (isspace(c))
            in_word = 0;
        else if (!in_word) {
            in_word = 1;
            n_words++;
        }
    }
    printf("%6d %6d %6d", n_lines, n_words, n_chars);
}

// process files given as arguments
// if no arguments process stdin
int main(int argc, char *argv[]) {
    if (argc == 1)
        process_stream(stdin);
    else
        for (int i = 1; i < argc; i++) {
            FILE *in = fopen(argv[i], "r");
            if (in == NULL) {
                fprintf(stderr, "%s: %s: ", argv[0], argv[i]);
                perror("");
                return 1;
            }
            process_stream(in);
            printf(" %s\n", argv[i]);
            fclose(in);
        }
    return 0;
}
```

grep.c

Over-simple /usr/bin/grep emulation.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// print lines containing the specified substring
// breaks on long lines, does not implement regexs or other grep features
void process_stream(FILE *stream, char *stream_name, char *substring) {
    char line[65536];
    int line_number = 1;
    while (fgets(line, sizeof line, stream) != NULL) {
        if (strstr(line, substring) != NULL)
            printf("%s:%d:%s", stream_name, line_number, line);
        line_number = line_number + 1;
    }
}

// process files given as arguments
// if no arguments process stdin
int main(int argc, char *argv[]) {
    if (argc == 2)
        process_stream(stdin, "<stdin>", argv[1]);
    else
        for (int i = 2; i < argc; i++) {
            FILE *in = fopen(argv[i], "r");
            if (in == NULL) {
                fprintf(stderr, "%s: %s: ", argv[0], argv[i]);
                perror("");
                return 1;
            }
            process_stream(in, argv[i], argv[1]);
            fclose(in);
        }
    return 0;
}
```

uniq.c

Over-simple /usr/bin/uniq emulation.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_LINE 65536

// cope stream to stdout except for repeated lines
void process_stream(FILE *stream) {
    char line[MAX_LINE];
    char lastLine[MAX_LINE];
    int line_number = 0;

    while (fgets(line, MAX_LINE, stdin) != NULL) {
        if (line_number == 0 || strcmp(line, lastLine) != 0) {
            fputs(line, stdout);
            strncpy(lastLine, line, MAX_LINE);
        }
        line_number++;
    }
}

// process files given as arguments
// if no arguments process stdin
int main(int argc, char *argv[]) {
    if (argc == 1)
        process_stream(stdin);
    else
        for (int i = 1; i < argc; i++) {
            FILE *in = fopen(argv[i], "r");
            if (in == NULL) {
                fprintf(stderr, "%s: %s: ", argv[0], argv[i]);
                perror("");
                return 1;
            }
            process_stream(in);
            fclose(in);
        }
    return 0;
}
```

count.c

```c
#include <stdio.h>
#include <string.h>

#define MAXLINE 80

int main(int argc, char **argv) {
    char line[MAXLINE];
    int count = 0;

    while (fgets(line, MAXLINE, stdin) != NULL) {
        if (strstr(line,"COMP2041") == line)
            count++;
    }
    printf("There are %d students in COMP2041\n", count);
    return 0;
}
```

anonymize_enrollments.py

```
import re, random, fileinput
from collections import defaultdict
enrollments = []
course_code_count= defaultdict(int)
seen = defaultdict(int)
first_name_count = defaultdict(lambda:defaultdict(int))
last_name_count = defaultdict(lambda:defaultdict(int))
for line in fileinput.input():
    enrollment =line.rstrip().split('|')
    enrollments.append(enrollment)
    (code,upi,name,program,plan,wam,session,birthdate,gender) = enrollment[0:9]
    course_code_count[code] += 1
    if upi not in seen:
        last_name = re.sub(r",.*", "", name)
        first_name = re.sub(r".*,\s*", "", name)
        first_name = re.sub(r"\s.*", "", first_name)
        first_name_count[gender][first_name] += 1
        last_name_count[gender][last_name] += 1
        seen[upi] += 1
for gender in first_name_count.keys():
    for (name,count) in  first_name_count[gender].items():
        if count < 3:
            first_name_count[gender].pop(name, 0)
    for (name,count) in  last_name_count[gender].items():
        if count < 3:
            last_name_count[gender].pop(name, 0)
upis = range(5000000,5100000)
enrolled_in = {}
student_details = {}
for enrollment in enrollments:
    (code,upi,name,program,plan,wam,session,birthdate,gender) = enrollment[0:9]
    if gender not in ['M', 'F']: continue
    if upi not in student_details:
        last_name = random.choice(last_name_count[gender].keys())
        if random.random() < 0.002:
            first_name = '.'
        else:
            first_name = random.choice(first_name_count[gender].keys())
            for i in range(0, 5):
                if random.random() < 0.25:
                    first_name += " " + random.choice(first_name_count[gender].keys())
        student_details[upi] = [str(upis.pop(random.randrange(len(upis)))), "%-50s"%((last_name+", "+first_name)[
```