ENEE 140 Lab 9

Lab instructions

This handout includes instructions for the recitation sessions on Wednesday and Friday. **Follow these instructions** to review to review the `switch`, `break` and `continue` statements, then **submit the homework** as indicated below.

1 Using `break` and `continue`

Modify the `break_continue.c` program from the class public directory on GRACE as follows:

1. Rewrite the two `for` loops with `while` loops, then with `do-while` loops.
2. What will be the output if you remove the first '`\n`' in the second and fourth `printf()` statements?
3. What will be the output if we use each of the following statement to replace the `for` loop

   ```
   for (x=1;x<10; x++)
   ```

   (a) `for (x=1; x<10; x=x+2)`

   (b) `for (x=2; x<10; x=x+2)`

   (c) `for (x=1; x!=10; x=x+2)`

   (d) `for (x=2; x!=10; x=x+2)`

2 `switch` practice

Rewrite the following code segment of `if-else` statements by using the `switch` statement. You can test your code by writing a complete C program. Make sure that you use representative values of a that cover all the possible cases.

```
if (a % 4 == 0) {
    printf("I am happy!\n");
} else {
    if (a % 4 == 2) {
        printf("I am not that happy.\n");
    } else {
        printf("I am sad. \n");
```

```
    }
}
```

3 Logical operators

Question 1: Verify De Morgan's laws by filling the missing values in the following truth table:

expr1	expr2	!(expr1 && expr2)	!expr1 || !expr2	!(expr1 || expr2)	!expr1 && !expr2
0	0				
0	1				
1	0				
1	1				

Question 2: Think about whether the following are true. If not, give a counter-example.

```
!(expr1 && expr2)  =  expr1 || expr2
(!expr1) && (!expr2) = !(expr1 || expr2)
```

Homework

Due: Friday at 11:59 pm.

Create two programs by following the instructions below.

Create a .zip archive containing your programs, then log into Elms, click on Gradescope in the course menu, then go to the relevant assignment to submit your work.

1 Caesar's code

Write a program, called `caesars_code.c`, to print your name using Caesar's code. Caesar's code is one of the earliest encryption algorithms and it works by substituting each letter with the letter that is 3 positions down the alphabet (for example, 'A' becomes 'D'). When shifting the letters from the end of the alphabet you must wrap around to the beginning (for example, 'z' becomes 'c'). Do not encrypt the space character. For example:

```
Name:           Tudor Dumitras
Encrypted name: Wxgru Gxplwudv
```

Hint. Using a loop, convert each character in the string to the corresponding character in Caesar's code. Pay attention to the difference between uppercase and lowercase characters.

2 Monthly calendar

The UNIX `cal` command prints out the calendar of the month/year that the user enters. Type in the following, one at a time, and observe the output:

```
cal 3 2014
cal 2014
cal 1 1
```

To learn more about this command, type in `man cal` for help.

Write a program named `cal.c` that prints out the following (which is the output when you type in `cal 4 2023` in UNIX). Note that you are not asked to implement the `cal` command. You can assume that you know April 1 is a Saturday. You will need to use loops and the % operator.

```
     April 2023
Su Mo Tu We Th Fr Sa
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

Reading assignment

K&R Chapters 7.1, 7.5, 7.6, 7.7, B1. Re-read chapters 7.2, 7.4.

Weekly challenge

Write a program that reads several file names from the command line, concatenates these files, and prints the result to the standard output.

You can use the following template (also available in the GLUE class public directory, at `public/challenges/week09`):

```c
/*
 * cat.c
 *
 *  Read several file names from the command line.
 *  Concatenate these files and print the result to the standard output.
 */

#include <stdio.h>


// Copy the content of one file to another; assume that the files are open.
// Return the number of characters copied
int
filecopy(FILE *in, FILE *out)
{
}

int
main(int argc, char *argv[])
{

        return 0;
}
```

```c
/*
 * cat.c
 *
 *  Read several file names from the command line.
 *  Concatenate these files and print the result to the standard output.
 */

#include <stdio.h>


// Copy the content of one file to another; assume that the files are open.
// Return the number of characters copied
int
filecopy(FILE *in, FILE *out)
{
}

int
main(int argc, char *argv[])
```

```
{

        return 0;
}
```

The weekly challenge will not be graded. However, if you manage to solve it, you may submit it for extra credit. The deadline for submitting your solution to the weekly challenge is **Monday at 11:59 pm**.

Submit it by going to the relevant assignment in Gradescope.