Laboratory 5: More on Program Selection: *if*, *if-else*, and *switch; and file redirection*

Lecture notes: Input redirect: a.out < input_file Output redirect: a.out >output_file Both: a.out < in >out

1. Basic syntax:

```
if (expression)
{ statements;
}
```

```
if (expression)
{ statements1;
}
else
{ statements2;
}
```

2. Selectively executing statements.
3. True or false: if the expression has a value 0, the condition/expression is considered to be false, otherwise it is considered to be true.
   a. A value or an arithmetic expression:

      True is the value or the result of the expression is non-zero.

      Example: *if (1)* is true all the time.

      if (a) is true when a≠0 and is false when a=0.

   b. Assignment: =

      Assignment (such as a = 3+2) also has a value which equals the value of the right handed side (5 in this case).

      Example: *if (a=3+2)* is true and assigns value 5 to a.

      if (b=0) is false and assigns value 0 to b.

   c. Simple relational operation: <, <=, >, >=, ==, *!=*

      One value, variable, or arithmetic operation is needed on each side of the operator. The expression is true if the relation holds.

      Example: 3.14 > 3.0 2*a != b grade = = 'A'

   d. Composed relational operation: *&&, ||, !*

      These are operations on logic operations.

      Example: (-1 < a) && (a < 1) (-1 >= b) || (1 <= b) !(a>2)

      Question: what is the simple expression of each of these logic operations

      (a <= 3) && (a >= 3) !(b != 4) !((c <= -2) || (c >=2))

4. Nested if, if-else
   a. See the sample code *ifelse.c* for examples.
   b. It is important to pair up the if's and else's. use {} when not sure.

5. switch statement:
    a. syntax:

```
switch(expression)
{ case value0: statements0; break;
  case value1: statements1; break;
...
 default:statements;
}
```

    b. Use int or char type for the expression.
    c. If the statements for multiple cases are the same, they can be combined.
    d. “default” case is normally used to handle errors.
    e. Sometimes it can be tricky to find the right expression.
6. Good programming style
    a. Follow normal English rules when possible (for example, in documentation and printf() statement) such as putting a space after comma.
    b. Leave one space on each side of a binary operator for readability.
    c. Indent code in a consistent fashion to indicate program flow.
    d. Place braces { } in a consistent fashion using one of the following two

```
i.  for ( i = 0; i < 100; i++) {
        statements;
    }
```

```
ii.  for ( i = 0; i < 100; i++)
    { statements;
    }
```

    e. For safety, in any statement body like those in loops and if, if-else, use { } even there is one statement.
    f. Give variables meaningful (and short) names when possible. When multiple words are used in variable name, use _ or letters in different cases to separate them.
    g. Whenever the program needs input from the user, first print out a message to prompt the user about what is expected.
    h. If input values need to meet certain conditions, check them the first thing when the values are read in.
    i. For operations that have some exceptions, e.g. b cannot be 0 in a / b, check such exceptions before the operation. (e.g. if (b != 0) …)
    j. Use simple statements, simple data structures, whenever possible. (e.g. sum = a++ + --b*2; might not be recommended.)
7. codes: ifelse.c, fileRedirect.c
8. Reading :textbook section 6.3

Name: _____ Section: 010__ Date: _____

Lab Report

These are for your practice in the recitation. Only submission is the quiz problem.
Work on your project 1, and complete quiz on Thursday.

1) What will the output of the following code segment? Check your answer by testing it in a complete program.

```
int a =5;
if (a = 0) { printf("a=0 is true. \n");}
else { printf("a=0 is not true.\n");}
if (a==0) { printf("a==0 is true. \n");}
else { printf("a==0 is not true.\n");}
```

2) (dangling else problem) What will the output of the following code segment with different values to x and y? Check your answer by testing it in a complete program.

```
int x,y
scanf("%d%d", &x, &y);;
if (x > 0)
    if (y>0)
        printf("Both positive.\n");
else
    printf("x negative, ignore y.\n");
```

a x = 2, y = 1

b x = 2, y = -1

c x = -2, y = -1

d x = -2, y = 1

What do you learn from this example?

3) Write a complete C code to read in one character from the keyboard and detect whether it is (i) a lower case letter, (ii) an upper case letter, (iii) a digit (0-9), or (iv) some other character.

4) Input redirect: ***a.out < input_file*** Output redirect: ***a.out >output_file*** Both: ***a.out < in >out***