Midterm Exam

Introduction to Computers and Programming

Summer 2000

Name ________________________________

Rules

• NO GROUP EFFORTS
• Closed books, closed notes, no calculators
• Follow directions carefully
• Please print (not a rule, just a suggestion)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Score /</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>
1. Below are some of the *Common Program Errors* found in the textbook:

2.3 Typing the name of the output function `printf` as `print` in a program.
2.4 Using a capital letter where a lowercase letter should be used.
2.6 Placing the calculation of an assignment expression to the left of the assignment operator (=).
2.16 Separating any of the symbols in the operators `==`, `!=`, `>=`, or `<=` with spaces.
2.17 Reversing any of the symbols in the operators `!=`, `>=`, or `<=`.
3.1 Forgetting one or both of the braces that delimit a compound statement.
3.2 Placing a semicolon after the condition in an `if-else` statement.
4.1 Using a floating point variable as a loop counter.
4.3 Using commas instead of semicolons in a `for` statement.

Each one of the following five code fragments contains exactly one of the above errors. **Identify the error in each code fragment.**

(a) 
```c
if (subject % 2 = = 0)
    printf("Even\n");
else
    printf("Odd\n");
```

(b) 
```c
while (studentNumber < classSize) {
    gradeTotal = gradeTotal + score;
    studentNumber = studentNumber + 1;
}
```

(c) 
```c
switch (score / 20) {
    case 5:
        printf("Great\n");
        break;
    case 4:
        printf("Good\n");
        break;
    default:
        printf("OK");
}
```

(d) 
```c
for (counter = 0, counter < 10, counter = counter + 1) {
    printf("How many? \n");
    scanf("%d", &apples);
}
```

(e) 
```c
do {
    a + b = c;
    x = y - z;
} while (e);
```
2. Consider the following code fragment:

```c
for (row = 1; row <= height; row = row + 1) {
    for (col = 1; col <= width; col = col + 1)
        printf("*");
    printf("\n");
}
```

(a) Assuming that the value of the variable `height` is 4, and that the value of the variable `width` is 5, what output is produced by this fragment?

(b) The fragment illustrated above uses a `for` loop nested inside of another `for` loop. Write a code fragment that produces the same output as the one above, but uses only one loop statement. Do not make any assumptions about the values of variables. *Hints:*

- This loop must be counter-controlled.
- The test expression of the loop must involve `height` and `width` as well as the counter variable.
- You need to place an `if` statement inside the body of the loop statement.
3. Determine the values of the following C expressions:

(a) 15 / 4
(b) 15 % 4
(c) 15.0 / 4.0
(d) 8 > 5 && 2 == 2
(e) (8 > 5) + 4
(f) 24 - 12 / 2 * 3 + 6
(g) 2 * (3 + (8 / 4))

4. Consider the following two code fragments. Fragment A is a part of a program presented in both class and the text; at first glance, fragment B appears to be almost identical to fragment A. In both fragments, assume that variables num1 and num2 are of type int.

**Fragment A:** uses simple if statements

```c
if (num1 == num2)
    printf("%d is equal to %d\n", num1, num2);
else if (num1 != num2)
    printf("%d is not equal to %d\n", num1, num2);
else if (num1 < num2)
    printf("%d is less than %d\n", num1, num2);
else if (num1 > num2)
    printf("%d is greater than %d\n", num1, num2);
else if (num1 <= num2)
    printf("%d is less than or equal to %d\n", num1, num2);
else if (num1 >= num2)
    printf("%d is greater than or equal to %d\n", num1, num2);
```

**Fragment B:** uses if-else statements

```c
if (num1 == num2)
    printf("%d is equal to %d\n", num1, num2);
else if (num1 != num2)
    printf("%d is not equal to %d\n", num1, num2);
else if (num1 < num2)
    printf("%d is less than %d\n", num1, num2);
else if (num1 > num2)
    printf("%d is greater than %d\n", num1, num2);
else if (num1 <= num2)
    printf("%d is less than or equal to %d\n", num1, num2);
else if (num1 >= num2)
    printf("%d is greater than or equal to %d\n", num1, num2);
```
(a) **Write the output produced by both fragments** assuming that the value of variable `num1` is 6 and the value of variable `num2` is 3.

(b) **Write the output produced by both fragments** assuming that the value of variable `num1` is 4 and the value of variable `num2` is 4.

(c) **Write the output produced by both fragments** assuming that the value of variable `num1` is 7 and the value of variable `num2` is 11.

(d) Obviously, the two code fragments do not produce the same output. Nevertheless, it is possible to write a chain of if-else statements that produces output identical to that of fragment A in all cases. **Write this code fragment.** *Hint: You need to use two condition expressions to select among three alternative actions.*