CS1010E Programming Methodology

Tutorial 3: Selection Statements

Learn as much as you can while you are young, since life becomes too busy later.

~ Dana Stewart Scott

To students:

Due to time constraint, not all the questions in this paper may be discussed in class. Please do go through those leftover questions after class.

Please cooperate with your tutors towards a fruitful and enriching learning experience.

I. Manual Tracing

1. Given three integers, we are supposed to find the maximum value among them.

(a) Student Adam wrote the following code fragment for the task.

```
// suppose num1, num2 and num3 are given by user input

int max = 0;
if (num1 > num2 && num1 > num3) {
  max = num1;
}
if (num2 > num1 && num2 > num3) {
  max = num2;
}
if (num3 > num1 && num3 > num2) {
  max = num3;
}
printf("Maximum value = %d\n", max);
```

Does the above code fragment work? If your answer is no, give a counter example (i.e. give three integers such that the above code fails to derive the correct answer).

(b) Student Brusco on the other hand, wrote the following code fragment.

```
// suppose num1, num2 and num3 are given by user input

int max = num1;
if (num2 > max) {
  max = num2;
} else {
  max = num3;
}
printf("Maximum value = %d\n", max);
```

Does the above code fragment work? If your answer is no, give a counter example.

(c) Student Richard wrote the following code fragment.

```
// suppose num1, num2 and num3 are given by user input

int max = 0;
if (num1 > max) {
  max = num1;
}
if (num2 > max) {
  max = num2;
}
if (num3 > max) {
  max = num3;
}
printf("Maximum value = %d\n", max);
```

Does the above code fragment work? If your answer is no, give a counter example.

(d) Give a correct code fragment that finds the maximum among 3 integer values.

2. **Conditional operator ? :** is sometimes used in place of the 'if-else' statement wherever appropriate. Study the following program for an example.

```
#include <stdio.h>

int main(void) {

  int n, p;
  printf("Enter an integer: ");
  scanf("%d", &n);

  p = ((n > 5) && (n < 20)) ? 33 : -77;

  printf("p = %d\n", p);

  return 0;
}
```

Do you know how to use the conditional operator now? Try to replace the 'if-else' statement in the following program with the conditional operator.

```
#include <stdio.h>

int main(void) {

  int a, b, max;
  printf("Enter 2 integers: ");
  scanf("%d %d", &a, &b);

  if (a > b) {
    max = a;
  } else {
    max = b;
  }

  printf("max = %d\n", max);

  return 0;
}
```

Replace the **if-else** statement in the box with the conditional operator.

3. Rewrite the following **switch** statement using **if-else** statements.

```
switch (grade) {
  case 10: case 9:
    a = 1;
    b = 2;
    break;
  case 8:
    a = 3;
    b = 4;
    break;
  default:
    a = 5;
    b = 0;
    break;
}
```

II. Hands-on Session

4. Download following program from cs1010e account:

```
cp ~cs1010e/tutorial/week4_q4.c .
```

Spot the errors in the program and correct them. Keep the function **func** below the **main** function.

```
#include <stdio.h>

int main(void) {

  void func(5);
  void func(3-7);

  return 0
}

void func(y) {
  if (y < 0) {
    printf("Nothing\n");
  } else {
    printf("Something\n");
  }
}
```

5. Download the following program from cs1010e account:

```
cp ~cs1010e/tutorial/week4_q5.c .
```

The program is replicated below.

```
// This program checks whether a user-input value
// is between 0 and 1, non-inclusive.

#include <stdio.h>

int main(void) {

  double value;

  printf("Enter value: ");
  scanf("%lf", &value);

  if (0 < value < 1) {
    printf("%f is between 0 and 1\n", value);
  } else {
    printf("%f is not between 0 and 1\n", value);
  }

  return 0;
}
```

(a) The **gcc** compiler issues a warning for the above program. If you ignore the warning and go ahead to run it, and enter 0.5 as the input, what result do you get? Why?

Conclusion: always compile your programs with the **–Wall** option, and do not ignore warnings. The compiler must have some good reason to alert you through its warnings.

(b) Suppose you correct the program in (a), but remove the ampersand (&) in the **scanf** statement. You ignore the compiler's warnings and go ahead to run the program. What will happen?

Tip: What can you do if you get a "core dump"?

Students often ask us, why does my program get a "core dump"? The answer is, there can be 101 reasons your program gets a "core dump", e.g., division-by-zero error or missing & in scanf, etc. Hence, it is IMPOSSIBLE for us to tell you right away what happened, unless we take a look at your program.

A “core dump” happens when a program terminates abnormally (crashes). The system then dumps the content of the memory into a file called “core”. This name comes from the old days when magnetic core memory was used.

The file “core” is usually very huge, so it is advisable that you delete it (use UNIX command “rm core”). What follows, of course, is that you have to study your program to find out what went wrong.

6. Download the following program from cs1010e account:

```
cp ~cs1010e/tutorial/week4_q6.c .
```

The program works but is not nicely written. Improve it by simplifying the logic and avoiding re-computation.

```
#include <stdio.h>

int main(void) {

  // declare the first input and second input
  double num1, num2;

  // ask user to enter two values
  printf("Enter two values: ");
  scanf("%lf %lf", &num1, &num2);

  if (num1/num2 < 90.2) {
    if (num1/num2 < 32.2) {
      printf("Paper\n");
    } else if (num1/num2 >= 45.8) {
      printf("Ruler\n");
    } else {
      printf("Pencil\n");
    }
  } else {
    if (num1/num2 >= 100.0) {
      printf("Unknown\n");
    } else if (num1/num2 < 100.0) {
      printf("Eraser\n");
    } else {
      printf("Clip\n");
    }
  }

  return 0;
}
```

Logic should be neat and clear.

7. **[Problem Set 1 Exercise #19]** Body Mass Index

Given a person's weight in kilograms and height in meters, his/her BMI (Body Mass Index) is calculated based on this formula:

$$\text{BMI} = \text{Weight} / \text{Height}^2$$

The following table shows the body types according to a person's gender and BMI:

	Female	Male
Underweight	BMI ≤ 19	BMI ≤ 20
Acceptable	BMI > 19 and ≤ 24	BMI > 20 and ≤ 25
Overweight	BMI > 24	BMI > 25

Write a program **bmi.c** to do the following:

1. Read the user's gender (type **int**), weight (type **double**) and height (type **double**).
2. Call a function **bodyType()** that takes in the above values, and returns the body type which is an integer.
3. Upon obtaining the body type, display a suitable advice for the user.

The gender is encoded using the following integers:

- 0 to represent female
- 1 to represent male

The body type is encoded using the following integers:

- -1 to represent underweight
- 0 to represent acceptable
- 1 to represent overweight

Three sample runs are shown below.

```
Enter your gender (0 for female, 1 for male): 0
Enter your weight (kg) and height (m): 62 1.6
Time to join the gym!
```

```
Enter your gender (0 for female, 1 for male): 1
Enter your weight (kg) and height (m): 62 1.6
Great! Maintain it!
```

```
Enter your gender (0 for female, 1 for male): 1
Enter your weight (kg) and height (m): 61.5 1.8
Stuff yourself with more food!
```