CS1010E Programming Methodology

Semester 1 2016/2017

Week of 3 – 7 October 2016

Tutorial 6

Functions as Procedures

1. The basis representation theorem states the following:

   > Every $n \in \mathbb{Z}^+$ can be uniquely expressed as a sum of terms $\langle a_i \rangle$ such that
   >
   > $$n = \sum_{i=0}^{k} a_i b^i, a_k \neq 0$$
   >
   > where b is the base of the number.

   For example, $(130)_{10}$ in base 10 can be expressed as $(1010)_5$ in base 5 since

   $$1 \times 10^2 + 3 \times 10^1 + 0 \times 10^0 = 1 \times 5^3 + 0 \times 5^2 + 1 \times 5^1 + 0 \times 5^0$$

   (a) Define a function `print10toB` that takes in a number n in the decimal (base-10) system and prints the equivalent number in base b, $2 \leq b \leq 10$.

   ```
   void print10toB(int n, int b);
   ```

   (b) Define a function `printB1toB2` that takes in a number n in base b_1 and prints the equivalent number in base b_2 with $2 \leq b_1, b_2 \leq 10$.

   ```
   void printB1toB2(int n, int b1, int b2);
   ```

2. Write a program that re-arranges three input integers in ascending order. The following `main` function is given to you.

   ```
   #include <stdio.h>

   int main(void) {
      int a, b, c;

      printf("Enter three numbers: ");
      scanf("%d%d%d", &a, &b, &c);
      printf("The numbers reordered: %d %d %d\n", a, b, c);
      return 0;
   }
   ```

   (a) Define a `swap` function that takes in two variables via function output parameters and performs the swap on them.

   (b) In the `main` function, call the `swap` procedure defined in 2a to reorder the three input in ascending order. You need not declare any other variables in `main`.

3. Given the month and the year, you are required to print the calendar for that month. As an example, the calendar for October 2016 is as follows:

```
 S  M Tu  W Th  F  S
                   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 31
```

As the solution to this problem is very complex, we will break down the problem into the following parts. You are advised to implement and test each part (by defining a suitable `main` function) before moving on to the next part.

(a) Define a `isLeapYear` function to determine if a given year is leap.

```
int isLeapYear(int year);
```

(b) Write a function `daysInMonth` to determine the number of days of a given month (expressed as an integer from 1 to 12) and year.

```
int daysInMonth(int month, int year);
```

Note that we will need to invoke the `isLeapYear` function from within the `daysInMonth` function in order to determine if a year is leap, and thus decide if February of that year has 28 or 29 days.

(c) Write a function `numDays` to compute the number of days elapsed on a given date since 1 January 1900.

```
int numDays(int day, int month, int year);
```

For example, `numDays(10,1,1900)` is 10, i.e. there are ten days between 1 January 1900 and 10 January 1900, both inclusive. As another example, `numDays(5,10,2012)` is 41186. Use the functions defined in question 3b.

(d) Write a function `dayOfWeek` to return the day of the week given the date.

```
int dayOfWeek(int day, int month, int year);
```

The days are represented by integer values ranging from 0 for Sunday, to 6 for Saturday. For example, `dayOfWeek(10,1,1900)` returns 3 (i.e Wednesday), and `dayOfWeek(18,2,1900)` returns 0 (i.e. Sunday). Note that 1 January 1900 falls on a Monday.

(e) Write a function `printMonth` to print the specific month of a given year.

```
void printMonth(int month, int year);
```

For example, the output for January 1900 is

```
  S  M Tu  W Th  F  S
     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 31
```

and the output for October 2016 is

```
  S  M Tu  W Th  F  S
                    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 31
```

Use the conversion specifier `"%3d"` to print an integer justified right within 3 spaces.

(f) Write a `main` function that repeatedly reads the month and year as input and displays the corresponding calendar. The program stops when a single zero is read as input. A sample run of the program is given in the following page. User input is underlined.

```
Enter the month and year: 10 2016
   S  M Tu  W Th  F  S
                     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 31

Enter the month and year: 2 2000
   S  M Tu  W Th  F  S
         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

Enter the month and year: 0
```