Exercises 10.1

Exercises 1-6 assume ASCII representation of characters and the following function f():

Algorithm 1: void f(char *ch*)

```
   /* This code is formatted differently from the book (and is not exactly C++ code) due to
      the program used to create this worksheet.  */
1  if (('A' ≤ ch) && (ch ≤ 'H')) then
2  |   f(ch - 1);
3  |   cout << ch;
4  else
5  |   cout << endl;
6  end
```

Tell what output will be produced by the function call.

1. f('C')

Solution:

f('C')

Line 1: The conditional on line 1 evaluates to **TRUE** so lines 2 and 3 are executed

Line 2: f('C'-1)=f('B')

  Line 1: The conditional on line 1 evaluates to **TRUE** so lines 2 and 3 are executed

  Line 2: f('B'-1)=f('A')

    Line 1: The conditional on line 1 evaluates to **TRUE** so lines 2 and 3 are executed

    Line 2: f('A'-1)=f('@')

      Line 1: The conditional on line 1 evaluates to **FALSE** so line 5 is executed

      Line 5: cout << endl *We indicate the new line with \n in the results*

    Line 3: cout << 'A'

  Line 3: cout << 'B'

Line 3: cout << 'C'

The resulting output:
\nABC

2. f('G')

Solution:

```
f('G')
    Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
    Line 2: f('G'-1)=f('F')
        Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
        Line 2: f('F'-1)=f('E')
            Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
            Line 2: f('E'-1)=f('D')
                Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                Line 2: f('D'-1)=f('C')
                    Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                    Line 2: f('C'-1)=f('B')
                        Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                        Line 2: f('B'-1)=f('A')
                            Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                            Line 2: f('A'-1)=f('@')
                                Line 1: The conditional on line 1 evaluates to FALSE so line 5 is executed
                                Line 5: cout << endl
                            Line 3: cout << 'A'
                        Line 3: cout << 'B'
                    Line 3: cout << 'C'
                Line 3: cout << 'D'
            Line 3: cout << 'E'
        Line 3: cout << 'F'
    Line 3: cout << 'G'
```

The resulting output:
\nABCDEFG

3. f('3')

Solution:

```
f('3')
    Line 1: The conditional on line 1 evaluates to FALSE so line is executed
    Line 5: cout << endl
```

The resulting output:
\n

Algorithm 2: void f(char *ch*)

```
   /* This code is formatted differently from the book (and is not exactly C++ code) due to
      the program used to create this worksheet.  */
1 if ((‘A’ $\le$ ch) && (ch $\le$ ‘H’)) then
2     f(ch + 1);
3     cout << ch;
4 else
5     cout << endl;
6 end
```

4. f(‘C’) if ch - 1 is replaced by ch + 1 in the function.

Solution:

The new algorithm is provided in Algorithm 2.

```
f(‘C’)
    Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
    Line 2: f(‘C’+1)=f(‘D’)
        Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
        Line 2: f(‘D’+1)=f(‘E’)
            Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
            Line 2: f(‘E’+1)=f(‘F’)
                Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                Line 2: f(‘F’+1)=f(‘G’)
                    Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                    Line 2: f(‘G’+1)=f(‘H’)
                        Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
                        Line 2: f(‘H’+1)=f(‘I’)
                            Line 1: The conditional on line 1 evaluates to FALSE so line 5 is executed
                            Line 5: cout << endl
                        Line 3: cout << ‘H’
                    Line 3: cout << ‘G’
                Line 3: cout << ‘F’
            Line 3: cout << ‘E’
        Line 3: cout << ‘D’
    Line 3: cout << ‘C’
```

The resulting output:
\nHGFEDC

Algorithm 3: void f(char *ch*)

```
   /* This code is formatted differently from the book (and is not exactly C++ code) due to
      the program used to create this worksheet.  */
1  if (('A' ≤ ch) && (ch ≤ 'H')) then
2      cout << ch;
3      f(ch - 1);
4  else
5      cout << endl;
6  end
```

5. f('C') if the output statement and the recursive call to f() are interchanged.

Solution:

The new algorithm is provided in Algorithm 3.

```
f('C')
    Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
    Line 2: cout << 'C'
    Line 3: f('C'-1)=f('B')
        Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
        Line 2: cout << 'B'
        Line 3: f('B'-1)=f('A')
            Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
            Line 2: cout << 'A'
            Line 3: f('A'-1)=f('@')
                Line 1: The conditional on line 1 evaluates to FALSE so line 5 is executed
                Line 5: cout << endl
```

The resulting output:
CBA\n

Algorithm 4: void `f`(char *ch*)

```
/* This code is formatted differently from the book (and is not exactly C++ code) due to
   the program used to create this worksheet.  */
1 if (('A' ≤ ch) && (ch ≤ 'H')) then
2 |   cout << ch;
3 |   f(ch - 1);
4 |   cout << ch;
5 else
6 |   cout << endl;
7 end
```

6. f('C') if a copy of the output statement is inserted before the recursive call.

Solution:

```
f('C')
    Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
    Line 2: cout << 'C'
    Line 3: f('C'-1)=f('B')
        Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
        Line 2: cout << 'B'
        Line 3: f('B'-1)=f('A')
            Line 1: The conditional on line 1 evaluates to TRUE so lines 2 and 3 are executed
            Line 2: cout << 'A'
            Line 3: f('A'-1)=f('@')
                Line 1: The conditional on line 1 evaluates to FALSE so line 5 is executed
                Line 5: cout << endl
            Line 4: cout << 'A'
        Line 4: cout << 'B'
    Line 4: cout << 'C'
```

The resulting output:
CBA\nABC

Determine what is calculated by the recursive functions in Exercises 11-15.

13.

Algorithm 5: unsigned `f`(unsigned n)

```
if ($n < 2$) then
    return 0;
else
    return 1 + f($n/2$);
end
```

Solution:

The floor of $\log_2 n$

14.

Algorithm 6: unsigned `f`(unsigned n)

```
if ($n == 0$) then
    return 0;
else
    return f($n/10$) + $n\%10$;
end
```

Solution:

The sum of the digits of n