EC327 – "Introduction to Software Engineering"

Lab3 – C++ Basics

Mo, 09/21/15 – Fri, 09/25/15

The goal of Lab3 is learning to approach solutions to defined problems using C++ under UNIX.

We define four problems (P1, P2, P3, P4) of similar complexity. To pass Lab3, **you choose one problem** and try to solve it. You are not required to solve all four problems!

The problems' solutions focus on the practical utilization of C++ **data types**, **user inputs, conditional branches (**`IF-ELSE`**)**, and **loops (**`DO-WHILE`**,** `FOR`**,** `WHILE`**).**

In your designated Lab hours you should demonstrate and explain your solution approach of the chosen problem. You are not required to work outside Lab hours! We recommend starting to work early. For example, you can solve the problem at home, come to your Lab hours, demonstrate your solution, and then you're done. But, ensure that your code compiles and executes properly in the Lab!

Read carefully the problem definitions. **Make your own decision** what problem to solve. **Decide for yourself** when to start. And please **do not hesitate to ask**!

GOOD LUCK!

P1 – “Diagonal” Pattern

diagonal.cpp

The user should **input** a non-negative integer number N that also must comply with the following constraints: `N>0` and `N<10`.

If the entered value is valid, then the program should print a diagonal line from right to left of the numbers from N to 1 (see Sample-Output below). For this purpose, use a loop (either `for` or `while`). In the loop, use the `cout` statement to print the diagonal line of numbers to the console.

If the entered value is not valid (`N<=0 or N>=10`), then print “`Invalid value for N!`” to the console (using `cout`) and the program should terminate. Hence, the program should run until the user enters an invalid value. Therefor use a `do-while` loop.

Sample Output / Test-Cases:

```
N: 1
1

N: 4
   4
  3
 2
1

N: 6
     6
    5
   4
  3
 2
1

N: 3
  3
 2
1

N: 10
Invalid value for N!
<Terminate>
```

P2 – “Square” Pattern

square.cpp

The user should **input** a non-negative integer number N.

If the entered number is less then 0 and greater equals 10 (`N<0 or N>=10`), then print “`Invalid value for N! Exiting.`” to the console (using `cout`) and the program should terminate. Hence, the program should execute until the user enters an invalid value. You can achieve this, using a `do-while` loop.

If the entered number equals 0 (`N==0`), then print “`Invalid value for N!`” to the console (using `cout`) and ask the user to enter a new number “`Please enter new number: `”. In this case, the program should not terminate.

If the entered value is valid (`N>0 or N<10`), then the program should print a square of size NxN of the numbers 1 to N (see Sample-Output below). For this purpose, use a loop (either `for` or `while`) and the `cout` statement to print the square to the console.

Sample Output / Test-Cases:

```
N: 0
Invalid value for N!

Please enter new number: 2
11
22

N: 5
11111
22222
33333
44444
55555

Please enter new number: 3
111
222
333

N: 10
Invalid value for N! Exiting.
```

P3 – Divisable by 3?

`div_by_three.cpp`

The user should **input** a non-negative integer number N (`N>0`)
We differentiate among the following cases of N:

- If the entered number is less than 0 (`N<0`), then output "`Please enter a number greater or equal to 0!`" to the console (using `cout`) and ask the user to input a new number.
- If the entered number is equal to 0 (`N==0`), then print "`Bye Bye!`" to the console (using `cout`) and the program should terminate.
- If the number is greater than 0 (`N>0`), then print all numbers from 1 to N that are divisible by 3 to the console (`i%3==0`).
- If there are no numbers divisible by 3, then output "`No numbers found!`" to the console.

Consequentially, the program should only terminate if the user enters 0. To achieve this behavior, you could use a `do-while` loop.

Sample Output / Test-Cases:

```
N: 5
3

N: 2
No numbers found!

N: -1
Please enter a number greater or equal to zero!

N: 10
3 6 9

N: 21
3 6 9 12 15 18 21

N: 0
Bye Bye!
```

P4 – Range

`range.cpp`

The user should **input** two integer numbers `a` and `b` with no constraints given.

If `a` is less then `b`, then print all numbers in **ascending order** from `a` to `b`. If `b` is less than `a`, then print all numbers in **descending order** from `a` to `b`. Print the numbers as a comma separated list.

Let the user input the numbers `a` and `b` until `a` equals `b` (`a==b`). Therefor use a `do-while` loop. If `a` equals `b`, then print "`Bye Bye!`" to the console and the program should terminate.

Sample Output / Test Cases:

```
a = 1
b = 5
1, 2, 3, 4, 5

a = 5
b = 1
5, 4, 3, 2, 1

a = -1
b = 1
-1, 0, 1

a = 1
b = 1
Bye Bye!
```