

# Assignment 5

CS 135: Computer Science I  
Spring 2008

## Objectives

1. Learn and demonstrate use of file I/O in C/C++
2. Learn and demonstrate use of relational operators in C/C++
3. Learn and demonstrate prototyping, defining, and calling simple value returning functions.

## 1 Golf Redux (7 points)

There are many ways to write a program that meets a specification. In this part of the assignment, rewrite (re-design, re-implement, and test) the golf program from assignment three **without** using functions (4 points). Call your source file `golfRedux.cpp`. Note that we have posted a solution to assignment three on the class web page. Now answer the following questions (Type in your answers, handwritten answers will not be accepted):

1. Which program is shorter (number of lines)? Why? (1 point)
2. If you made a mistake in the the formula for computing time, how many times would you correct it in `golfRedux.cpp`? How many times in the original golf program from assignment three? Why? (1 point)
3. Which of the two programs is better designed? Why? (1 point)

We expect one or two paragraphs of writing to answer **each** of the **Whys**. For this part, in addition to source code and program output, make sure that you hand in the answers to these questions on a separate sheet of paper. Sample runs will look exactly like the ones for assignment three.

## 2 Counting Red Blood Cells (13 points)

Red Blood Cells (RBCs) are the most common type of blood cell and the vertebrate body's principal means of delivering oxygen from the lungs or gills to body tissues via the blood. Figure 1 shows some. In a small drop of blood ( $\sim 1\mu$  litre) there are 4.7 – 6.1 million RBCs in men, 4.2 – 5.4 million RBCs in women, and 4.6 – 4.8 million RBCs in children. Too few red blood cells implies that you may be **anemic** and reduces the delivery of oxygen ( $O_2$ ) to tissues. Too many red blood cells implies **polycythemia** and can increase blood volume, pressure, and viscosity.

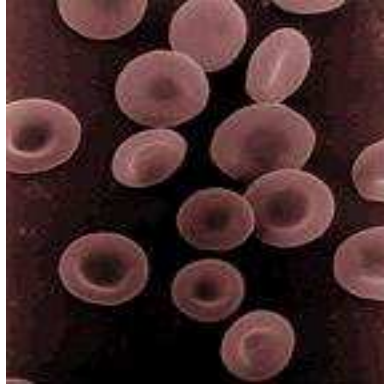


Figure 1: Human Red Blood Cells

Now given an image like the one in Figure 2 below, we can use artificially intelligent computer vision techniques to count the number of RBCs seen through a microscope in the triple-lined square. Suppose Leandro Loss (one of our TAs), who does research in computer vision, has written a program that reads images like Figure 2 and outputs a **file** with the following information in it:

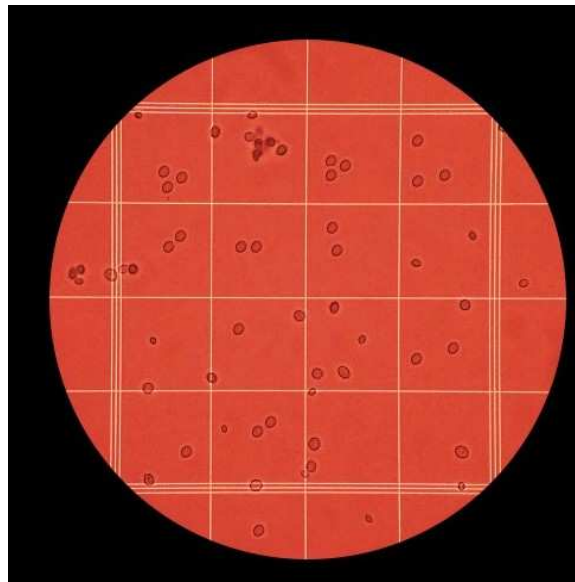


Figure 2: Number of RBCs in subject0's blood sample as seen through a microscope

- Subject identifier
- Subject gender
- Subject age
- Number of RBCs in  $\frac{1}{100000} \mu$  litre.

Design, implement, and test a complete C/C++ program that reads this data from the file and outputs whether the patient has anemia, is normal, or has polycythemia. Now that you know why functions are useful, ensure that you prototype, define, and use **at least two** functions.

Here's an example: suppose the input file `subject0.txt` contains:

```
Smith
male
32
58
```

Then the patient `Smith` is an `adult` (over 18), `male`, with 5.8 million RBCs per  $\mu$  litre of blood. The normal number of RBCs for an adult male is 4.7 – 6.1 million RBCs. `Smith's` RBC count falls well within this range, so she or he has a **normal** RBC count. Here's sample interaction with the program (`rbc` is the executable name):

```
sushil@xpc:~/samba/classes/135/assignments/as5/code$ ./rbc
Enter subject's filename
subject0.txt
```

```
-----
Subject name: Smith
      gender: male
          age: 32
          has: 5.8e+06 Red Blood Cells per micro litre of blood
```

Smith has a normal RBC count

```
sushil@xpc:~/samba/classes/135/assignments/as5/code$ ./rbc
Enter subject's filename
subject1.txt
```

```
-----
Subject name: Melon
      gender: female
          age: 53
          has: 1.66e+07 Red Blood Cells per micro litre of blood
```

Melon has polycythemia

```
sushil@xpc:~/samba/classes/135/assignments/as5/code$ ./rbc
Enter subject's filename
subject20.txt
Error in opening file: subject20.txt
Exiting....
```

`subject0.txt` and `subject1.txt` are linked from the class web page.

**Always** check for errors when opening input and output files. We suggest ensuring that your input file is in the same folder as your program executable. Although the class web page has a couple of test input subject files for you to use, we will test your program on test input files that

are not provided to you. As part of this assignment, create input files that thoroughly test your program. Use `vi`, `emacs`, `xemacs`, `wordpad`, or `notepad` to create your input files. Do not use MSWord.

### 3 Handing it in

Turn in a Folder (Binder) containing:

1. Cover sheet with
  - (a) Assignment Number
  - (b) Section Number
  - (c) Your name
  - (d) Your email
  - (e) Your TA's name
2. Source code (.cpp) file (s) on a CD/USB stick with your name and section number written on your CD/USB stick.
3. Executables for all parts of the assignment on the CD/USB stick.
4. For part 1, a printout of your numbered answers to the questions.
5. **Hardcopy of your source code (printout)**
6. A printout of sample runs of your program on a good set of test cases.

Ask an instructor or TA if you have questions.