COMP1511 17s1 (http://www.cse.unsw.edu.au/~cs1511/17s1/)

Assignment 1 - CAPTCHA Cracking

Introduction to Programming (http://www.cse.unsw.edu.au/~cs1511/17s1/)

CAPTCHA Explained

Your task in this assignment to write a C program which automatically recognizes the digits in a CAPTCHA (https://en.wikipedia.org/wiki/CAPTCHA) image.
A CAPTCHA is an attempt to determine whether or not a user is human. It is in effect a reverse Turing test (https://en.wikipedia.org/wiki/Turing_test). CAPTCHA are designed to be difficult to recognize with a computer - the design and assessment of this assignment recognizes this difficulty.

The input to your program will be a black-and-white (monochrome) image in a simple format described below. Each image will contain either 1 or 4 digits. The output of your program should be the digits in the images

Image Format

Common image formats such as JPEG (https://en.wikipedia.org/wiki/JPEG) and PNG (https://en.wikipedia.org/wiki/Portable_Network_Graphics) are complex, and decoding them would too difficult a task for this assignment.
Instead this assignment uses portable bitmap format (https://en.wikipedia.org/wiki/Netpbm_format) (PBM) for images. This is very simple ASCII format. The first line of each file will contain the characters "P1" identifying its format. The next line will contain 2 integers, the width and height of the images. The remainder of the lines in the file contain '1's and '0's specifying the pixel values of the image. Here is an example (digit/2_06.pbm)

Here is **read_pbm.c** (https://cgi.cse.unsw.edu.au/~cs1511cgi//assignments/captcha/read_pbm.c) which contains a function to read PBM files:

```
int read_pbm(char filename[], int height, int width, int pixels[height][width]);
```

It is strongly recommended you use **read_pbm** in your assignment rather than writing your own code.

Part 1 - Digit Cracking

The first part of this assignment is to write a C program **crack_digit.c**.
crack_digit will be given one command line argument, an image filename.

The image will contain a single digit.

You program should print only a single line of output

This line of output should contain only the digit in the images.

For example:

```
$ ./crack_digit digit/3_42.pbm
3
$ ./crack_digit digit/7_99.pbm
7
$ ./crack_digit digit/0_12.pbm
0
```

A dataset of 1000 example digit images (digit.html) is available to help you develop your program.
The week 7 lab exercises take you through getting started on **crack_digit.c**.

Part 2 - CAPTCHA Cracking

The second part of this assignment is to write a C program **crack_captcha.c**.
crack_captcha will be given one command line argument, an image filename.

The image will contain 4 digits.

You program should print only a single line of output

This line of output should contain only the 4 digit in the images.

For example:

```
$ ./crack_captcha captcha/4224.pbm
4224
$ ./crack_captcha captcha/9264.pbm
9264
$ ./crack_captcha captcha/0053.pbm
0053
```

A dataset of 1000 example captcha images (captcha.html) is available to help you develop your program.

Challenge CAPTCHA Cracking

The challenge part of this assignment is to identify more difficult captcha images with **crack_captcha.c**.
A dataset of 1000 example challenge captcha images (captcha_challenge.html) is available to help you develop your program.

Testing

The script `~cs1511/bin/captcha_test` will automatically test your programs on a random subset of a specified size of the supplied images:

```
$ ~cs1511/bin/captcha_test --digit -n 10 crack_digit.c captcha.h other_C_files
dcc crack_digit.c read_pbm.c -o crack_digit
dcc crack_digit.c read_pbm.c --valgrind -o crack_digit-valgrind
Running 10 tests
Test digit/5_95.pbm passed
...
$ ~cs1511/bin/captcha_test --captcha -n 20 crack_captcha.c captcha.h other_C_files
dcc crack_captcha.c read_pbm.c --valgrind -o crack_captcha-valgrind
Running 20 tests
Test captcha/8119.pbm passed
...
$ ~cs1511/bin/captcha_test --challenge -n 30 crack_captcha.c captcha.h other_C_files
cc crack_captcha.c read_pbm.c -o crack_captcha
dcc crack_captcha.c read_pbm.c --valgrind -o crack_captcha-valgrind
Running 30 tests
Test captcha_challenge/1936.pbm passed
...
```

Hints

You should follow discussion about the assignment in the class forums. Questions about the assignment should be posted there so all students can see the answer.
Don't panic!

Don't expect digit or capture identification to be perfect, just identify as many images as possible correctly.

The week 7 lab exercises showed you how to use one attribute (horizontal balance) to separate some images giving you a program that recognizes 20% of digits. Check out the sample solutions when they are released immediately after the lab is due.

Think about other digit attributes you might calculate.

Here are some possibilities that aren't too hard to calculate:

Tallness	*height*/*width* of the bounding box
Density	fraction of pixels in the bounding box that are black
Vertical balance	vertical equivalent of horizontal balance
Holes	number of holes in the image
Hole Fraction	area of white pixels in holes as fraction of bounding box

There are more possibilities (and no right way to approach this).

If you invent attributes try to make them not depend on the size of the digit (scale invariant).

Assumptions/Restrictions

You submitted code must be C only. You may not submit code in other languages. You may not use *system* or other C functions to run external programs.
You may submit data files with the suffix .txt to be used by your program.

You may call functions from the standard C library and the maths library. You may not use functions from other libraries - in other words your program must compile with dcc without the -l flag being used to specify an extra library. If you don't understand this requirement don't worry - as long as the autotest scripts compile your program you are fine.

You program must take at most 30 seconds to process an image on a CSE computer when compiled with dcc.

You may assume digit images are 70 pixels high and 50 pixels wide.

You may assume captcha images are 72 pixels high and 200 pixels wide and challenge captcha are 70 pixels high and 200 pixels wide.

You can assume there is one and only one digit in the digit images and only 4 digits in the captcha images.

You can assume digits are roughly vertically oriented, in other words the right-way up more-or-less.

You can assume digits are a similar size to the digits in the supplied test images.

You can not assume that the digits do not touch the edge of the image.

Otherwise, make as few assumptions as you can about the images. In particular, you should try **not** to make assumptions about the exact pattern of pixels used for a particular digit.

The images used to test your programs will be different to the images you have been supplied.

You can however assume there will be no major difference in the depiction of digits. The test images give a reasonable indication of the type of variation in the depiction of digits that your program should handle.

Submission of Work

You are required to submit intermediate versions of your assignment.
Every time you work on the assignment and make some progress you should copy your work to your CSE account and submit it using the give command below.

It is fine if intermediate versions do not compile or otherwise fail submission tests.

Only the final submitted version of your assignment will be marked.

All these intermediate versions of your work will be placed in a git repo and made available to you via a web interface at this URL, replace *z5555555* with your zpass. `https://gitlab.cse.unsw.edu.au/z5555555/17s1-comp1511-ass1/commits/master`

This will allow you to retrieve earlier versions of your code if needed.

You submit your work like this:

```
$ give cs1511 ass1 crack_digit.c crack_captcha.c captcha.h  other files
```

You may submit other **.c** or **.h** files.
When **crack_digit.c** and **crack_captcha.c** are compiled all other submitted C files will be compiled with them.

This will allow you define functions for use in both **crack_digit.c** and **crack_captcha.c**.

It is fine if these file contain functions used only by one of **crack_digit.c** or **crack_captcha.c**. Unused functions will not affect the compilation of the other program.

Only **crack_digit.c** and **crack_captcha.c** should contain main functions.

No other function name should be used twice.

You do not need to submit **read_pbm.c**. It will be automatically compiled with your programs. It is strongly recommended you use **read_pbm.c** unchanged. If you ignore this advice, do not create functions with the same names as functions in **read_pbm.c**

Blogging

You must blog every time you work on this assignment, recording how much time you spent working on the assignment and what this time was spent doing (reading, designing, coding, testing, debugging, ...).
You must blog about all significant bugs in your assignment including what test found the bug, how the bug was tracked down and fixed, how long this took and any lessons learnt.

You may create one big blog post and edit each time, or multiple small blog posts for the assignment.

Attribution of Work

This is an individual assignment. The work you submit must be your own work and only your work apart from these exceptions.
Joint work is not permitted.
You may use code written with your lab partner for the functions in the week 7 lab. This code should include comments indicating the joint-authorship with your lab partner.

You may use code from the sample solutions for the week 7 lab. You should add comments indicating the source of this code.

You may use small amounts (< 10 lines) of general purpose code (not specific to the assignment) obtained from site such as Stack Overflow or other publically available resources. You should attribute clearly the source of this code in a comment with it.

You are not permitted to request help with the assignment apart from in the course forum or from course staff.

Do not provide or show your assignment work to any other person (including by posting it on the forum) apart from the teaching staff of COMP1511.

The work you submit must otherwise be entirely your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined both automatically and manually for such submissions.

Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent. This may apply even if your work is submitted by a third party unknown to you.

Note, you will not be penalized if your work is taken without your consent or knowledge.

Assessment

This assignment will contribute 15% to your final mark.

The assessment for the assignment recognizes the difficulty of the task.

5% of the marks for assignment 1 will come from your tutor reading your blog. They will assess generously any genuine attempts to record the information described above.

15% of the marks for assignment 1 will come from hand marking of the readability of the C you have written. These marks will be awarded on the basis of clarity, commenting, elegance and style. In other words, your tutor will assess how easy it is for a human to read and understand your program.

80% of the marks for this assignment will be based on the performance of your program on a set of images which you have not seen.

Here is an indicative marking scheme.

90+%	recognizes many challenge captchas, beautiful code, well blogged
80%	recognizes 90+% of captchas correctly, very readable code, well blogged
70%	recognizes 90+% of digits, readable code, well blogged
55%	recognizes 50+% of digits, decent code, blogged
-70%	Knowingly providing your work to anyone and it is subsequently submitted (by anyone).
-70%	Submitting any other person's work. This includes joint work.
0 FL for COMP1511	Paying another person to complete work. Submitting another person's work without their consent.

The lecturer may vary the assessment scheme after inspecting the assignment submissions but it will remain broadly similar to the description above.

Due Date

This assignment is tentatively due at 23:59pm Sunday Apr 30.
If your assignment is submitted after this date, each hour it is late reduces the maximum mark it can achieve by 2%. For example if an assignment worth 74% was submitted 10 hours late, the late submission would have no effect. If the same assignment was submitted 15 hours late it would be awarded 70%, the maximum mark it can achieve at that time.