C Coding Conventions

1. Introduction

This document is intended to outline the C coding conventions for the homeworks and labworks of the classes CSE102 and CSE108, respectively. These conventions will provide a standard and consistent look and feel to the source code generated by students. This document will also serve as a template and reference for source code files. This should ensure that required features like necessary comment blocks are included in all files.

The conventions in this document must be followed by all students and will be enforced with a penalty of 10% of the total grade for a homework or labwork.

For all examples in this document, a student with the number 091041004 is considered. Replace with your student number when working with your files.

Also some helpful tips for gedit are given to ease the following of these conventions. Students are encouraged to use gedit as their default editor when writing C source codes.

2. File Naming

- For homeworks in CSE102, file names will be restricted to the following format: **HW##_<StudentNumber>_part#.c** (same for .h files) (replace ## and # with the proper numbers)

- For labworks in CSE108, file names will be restricted to the following format: **LW##_<StudentNumber>_part#.c** (same for .h files) (replace ## and # with the proper numbers)

  For example, if the homework 5 of CSE102 has two parts, then the source code files will be named like this:
  HW05_091041004_part1.c
  HW05_091041004_part2.c

- If a homework or labwork does not have multiple parts, part# can be omitted from the filename.

  For example, if the labwork 3 of CSE108 has only one part to submit, then the source code will be named like this:
  LW03_091041004.c

3. Source File Formats (Both .c and .h)

- A line of text in a source file must not exceed 80 characters. This requirement is intended to prevent the text from wrapping.

In gedit, go to *Edit->Preferences*. In the *View* tab, select *Display right margin* and choose column 80. This will help you by showing the 80-th column with a vertical line, so that you may break a line before the 80-th column.

- Indenting with in the source file will be 4 spaces. If tabs are used, they should be converted to 4 spaces before the source code is sent.

  In gedit, the tab key on the keyboard can be directly used to put 4 spaces. To enable this option, go to *Edit->Preferences*. In the *Editor* tab, enter *Tab width* as 4 and select *Insert spaces instead of tabs*.

- A source file (.c file) will follow the outline listed below.

  - File comment box
    - Name of the file
    - Creation date
    - Author's name and number
    - Description of the file
    - Notes (if any)
    - Referenced technical information
  - Include section
  - Constant definition section (#defines goes here)
  - Type definition section (typedefs goes here)
  - Macro section
  - Global variable declaration section
  - Function prototype section for static functions (a comment box for each function)
    - Function name
    - Definition of parameters
    - Definition of the return value
    - Description of the function
    - Notes (if any)
  - Function section
    - Brief description of functions
    - Function bodies (code of functions)
  - End of file comment

- A source file (.h file) will follow the outline listed below.

  - File comment box
    - Name of the file
    - Creation date
    - Author's name and number
    - Description of the file
    - Notes (if any)
    - Referenced technical information

- #ifndef section to prevent multiple includes
- Include section
- Constant definition section (#defines goes here)
- Type definition section (typedefs goes here)
- Macro section
- Global variable declaration section (if necessary)
- Function prototype section (a comment box for each function)
  - Function name
  - Definition of parameters
  - Definition of the return value
  - Description of the function
  - Notes (if any)
- #endif (close #ifndef)
- End of file comment

4. Coding

- ***Constants***
  All upper case letters. Use _ to separate words.
  ```
  #define MAX_RATIO 0.5
  ```

- ***Parameterized macros***
  All upper case letters. Use _ to separate words. Use parenthesis to protect arguments.
  ```
  #define MIN(a, b)     ( (a) < (b) ? (a) : (b) )
  ```

- ***Types***
  Capitalize each word in the type name. Use only letters and numbers, no underscores.
  ```
  Pixel
  Byte8
  FractionalNumber
  LinkedListNode
  ```

- ***Variables***
  Start with lower case and capitalize each following word. Use only letters and numbers, no underscores. Avoid choosing meaningless variable names, such as `a`, `b`, `k`, unless it is a loop counter.
  ```
  maxSoFar
  repeatCount
  priority
  ```

  For pointer variables, use the suffix Ptr
  ```
  listPtr
  myListPtr
  ```

  For global variables, use the suffix Gbl
  ```
  sinTableGbl
  ```

- ***Functions***

Capitalize each word, Use only letters and numbers, no underscores.

```
SendMessage(...)
CreateList(...)
Insert(...)
```

5. Indentation

- ***Functions***

  The indentation style for functions looks as follows:

```
<return-type> <function-name> ( <parameters> )
{
      <declerations>
      <body>
}
```

  <declerations>

  Define each variable in a separate line and comment each variable on its line.

```
double average;   /* Average grade of the students */
int studentCount; /* Total number of students */
```

  <parameters>

  Variable naming in the previous section applies to parameters.

- ***if***

  The indentation style for `if` blocks looks as follows:

```
if ( <condition> )
{
      <code>
}
else if ( <condition> )
{
      <code>
}
else
{
      <code>
}
```

  Even if the <code> part is single line, { } brackets must be used.
  If the closing } bracket is separeted from the opening { bracket by a distance not easily displayed, then the closing } bracket should identify the initial statement.

```
if( <condition> )
{
      ...
      <a lot of lines of code>
      ...
} /* End of if( <condition> ) */
```

- ***while***

  The indentation style for `while` blocks looks as follows:

```
while( <condition> )
{
      <code>
}
```

Even if the <code> part is empty, { } brackets must be used.
If the closing } bracket is separeted from the opening { bracket by a distance not easily displayed, then the closing } bracket should identify the initial statement.

```
while( <condition> )
{
      ...
      <a lot of lines of code>
      ...
} /* End of while( <condition> ) */
```

- ***do while***
  The indentation style for `do while` blocks looks as follows:

```
do
{
      <code>
}
while( <condition> );
```

Even if the <code> part is empty, { } brackets must be used.
If the closing } bracket is separeted from the opening { bracket by a distance not easily displayed, then the opening { bracket should identify the conditional statement.

```
do /* Start of while( <condition> ) */
{
      ...
      <a lot of lines of code>
      ...
}
while( <condition> );
```

- ***for***
  The indentation style for `for` blocks looks as follows:

```
for(<init>; <condition>; <update>)
{
      <code>
}
```

Even if the <code> part is empty, { } brackets must be used.
If the closing } bracket is separeted from the opening { bracket by a distance not easily displayed, then the closing } bracket should identify the initial statement.

```
for(<init>; <condition>; <update>)
{
      ...
      <a lot of lines of code>
      ...
```

```
} /* End of for( <condition> ) */
```

- ***switch***
  The indentation style for `switch` blocks looks as follows:

```
switch(<expression>
{
      case <const>:
            {
                  ...
                  <a lot of lines of code>
                  ...
            } /* End of case <const> */
            break;

      case <const>:
            <code>
            /* Fall through */

      case <const>:
            <code>
            break;

      ...

      default:
            <code>
}
```

  All which statements must have a default case. If the default case is empty, comment the default case as `/* do nothing */`
  All case statements which allow fall through, must be commented.
  If a case statement is long, it should be bracketed { }, and if brackets are far apart, then the closing } bracket must have a comment referring to the original case statement;

6. Example .h file

```
/*############################################################################*/
/*                                                                            */
/* HW05_091041004_part1.h                                                     */
/* ---------                                                                  */
/* Created on 28/02/2010 by Alparslan YILDIZ                                  */
/*                                                                            */
/* Description                                                                */
/* -----------                                                                */
/*    The HW05_091041004_part1.h contains functions for the first part of     */
/*    the homework II, which are used for drawing a shape.                    */
/*                                                                            */
/* Notes                                                                      */
/* -----                                                                      */
/* This is a demo files. Usually, the text with in the prolog is more         */
/* verbose.                                                                   */
/*                                                                            */
/* References                                                                 */
/* ----------                                                                 */
```

```
/* (If any)                                                                  */
/*                                                                           */
/*###########################################################################*/

#ifndef _HW05_091041004_PART1_H_
#define _HW05_091041004_PART1_H_

/*---------------------------------------------------------------------------*/
/*                                Includes                                   */
/*---------------------------------------------------------------------------*/
#include <stdio.h>

/*---------------------------------------------------------------------------*/
/*                                 #defines                                  */
/*---------------------------------------------------------------------------*/
#define    PI               3.14
#define    MAX_EDGE_LEN       16

/*---------------------------------------------------------------------------*/
/*                                  typedefs                                 */
/*---------------------------------------------------------------------------*/
typedef unsigned char        Byte;
typedef struct
{
    Byte        pix;
    int         depth;
}Pixel;

/*---------------------------------------------------------------------------*/
/*                         Function Prototypes                               */
/*---------------------------------------------------------------------------*/

/*###########################################################################*/
/*                                                                           */
/* int DrawSquare(int ox, int oy, int a)                                     */
/* -----------                                                               */
/*                                                                           */
/*     ox - x coordinate of the center of the square                         */
/*     oy - y coordinate of the center of the square                         */
/*      a - Edge length of the square                                        */
/*                                                                           */
/* Return                                                                    */
/* ------                                                                    */
/*     0 on success                                                          */
/*     -1 on error                                                           */
/*                                                                           */
/* Description                                                               */
/* -----------                                                               */
/*     This function takes the center and edge length of a square and draws  */
/*     it on the screen if the center coordinates are inside the screen and  */
/*     edge length does not exceeds screen limits.                           */
/*                                                                           */
/* Notes                                                                     */
/* -----                                                                     */
/*     The user is responsible that the screen is clear at the time of call. */
/*                                                                           */
/*###########################################################################*/
int DrawSquare(int ox, int oy, int a);
```

```
#endif

/*############################################################################*/
/*                        End of HW05_091041004_part1.h                       */
/*############################################################################*/
```

7. Example .c file

```
/*############################################################################*/
/*                                                                            */
/* HW05_091041004_part1.c                                                     */
/* ---------                                                                  */
/* Created on 28/02/2010 by Alparslan YILDIZ                                  */
/*                                                                            */
/* Description                                                                */
/* -----------                                                                */
/*    The HW05_091041004_part1.c contains the implementations of functions    */
/*    for the first part of the homework II, which are used for drawing       */
/*    a shape.                                                                */
/*                                                                            */
/* Notes                                                                      */
/* -----                                                                      */
/* This is a demo files. Usually, the text with in the prolog is more         */
/* verbose.                                                                   */
/*                                                                            */
/* References                                                                 */
/* ----------                                                                 */
/* (If any)                                                                   */
/*                                                                            */
/*############################################################################*/

/*----------------------------------------------------------------------------*/
/*                                 Includes                                   */
/*----------------------------------------------------------------------------*/
#include <stdlib.h>
#include "HW05_091041004_part1.h"

/*----------------------------------------------------------------------------*/
/*                          Function Prototypes                               */
/*----------------------------------------------------------------------------*/

/*############################################################################*/
/*                                                                            */
/* int DrawPixel(int x, int y, char pen)                                      */
/* -----------                                                                */
/*                                                                            */
/*      x   - x coordinate of the pixel                                       */
/*      y   - y coordinate of the pixel                                       */
/*      pen - the character pen will be used to draw the pixel                */
/*                                                                            */
/* Return                                                                     */
/* ------                                                                     */
/*     0 on success                                                           */
/*     -1 on error                                                            */
/*                                                                            */
```

```
/* Description                                                              */
/* -----------                                                              */
/*     This function takes the position of a pixel and draws it on the      */
/*     screen if the pixel coordinates are inside the screen and pen is a   */
/*     valid character.                                                     */
/*                                                                          */
/****************************************************************************/
static int DrawPixel(int x, int y, char pen);


/*--------------------------------------------------------------------------*/
/*                          Function Implementations                        */
/*--------------------------------------------------------------------------*/

/* Function DrawPixel                                                       */
/* -----------                                                              */
/*     This function takes the position of a pixel and draws it on the      */
/*     screen if the pixel coordinates are inside the screen and pen is a   */
/*     valid character.                                                     */
static int DrawPixel(int x, int y, char pen)
{
    ...
    <code of the function>
    ...
}

/* Function DrawSquare                                                      */
/* -----------                                                              */
/*     This function takes the center and edge length of a square and draws */
/*     it on the screen if the center coordinates are inside the screen and */
/*     edge length does not exceeds screen limits.                          */
int DrawSquare(int ox, int oy, int a)
{
    ...
    <code of the function>
    ...
}


/****************************************************************************/
/*                       End of HW05_091041004_part1.c                      */
/****************************************************************************/
```