cc-practice-functions

Table of Contents

1. README

- Practice workbook for functions in C

2. TODO Identify yourself

- replace the placeholder `[yourName]` in the header of this file by your name and save the file (`C-x C-s`).

3. TODO `hello_world` function

1. The `hello_world` function does not have a `return` value, and it takes no arguments. Complete the code in 1 so that it compiles and runs.

2. Remember that all statements that you want to execute need to be contained in the body of the `main` function.

```
// function definition
___ hello_world(___)
{
  printf("Hello world\n");
}
// function call
hello_world();
```

Solution

```
// function definition
int hello_world(void)
{
  printf("Hello world\n");
  return 0;
}
// function call
hello_world();
```

```
Hello world
```

4. TODO `main` and `hello_world`

1. Create a complete C program `hello.c` **without** tangling an Org-mode file[1]. Use GNU Emacs as C source code editor: create the file with `C-x C-f hello.c RET`
2. The file should contain a `main` function, the `hello_world` function definition, and a function call.
3. Compile and run the file on the command line.

Solution

```c
#include <stdio.h>

// function definition
int hello_world(void) {
  puts("hello world");
}

// main function with function call
int main(void) {
  hello_world();
  return 0;
}
```

```
hello world
```

5. TODO Save a function return value

1. Run the `average` function in 1 below, save and print its value.

   Sample output:

   ```
   : The average of 5.1 and 8.9 is: 7
   ```

2. The function is already defined at the top of the code block, and two `double` values are declared and defined, too.

3. *Tip: remember to declare your variable. Use ~%g~ for the output.*

   ```
   // function definition
   double average (double a,double b){return (a+b)/2;}

   // Input variable declaration and definition
   double x = 5.1, y = 8.9;

   // Save the average of x and y in a variable avg
   ____
   // Print the variable avg
   ____
   ```

Solution

```
// function definition
double average (double a,double b){return (a+b)/2;}

// Input variable declaration and definition
double x = 5.1, y = 8.9;

// Save the average of x and y in a variable avg
double avg = average(x,y);

// Print the variable avg
printf("The average of %g and %g is: %g\n", x, y, avg);
```

```
The average of 5.1 and 8.9 is: 7
```

6. TODO Write your own function

1. Write a function `add` that adds two integer numbers `num1` and `num2`.
2. Call the function `add` inside the `printf` function for the sample arguments `i=100` and `j=200`.

   Sample output:

   ```
   : 100 + 200 = 300
   ```

   ```
   // function definition

   // function call and print

   ```

Solution

```
// function definition
int add(int i, int j) {
  return (i + j);
}

// declare and initialize
int k = 100, l = 200;

// function call and print
printf("%d + %d = %d\n", k , l, add(k,l) );
```

```
100 + 200 = 300
```

```
100 + 200 = 300
```

7. TODO Use a simple function

1. Put the definition of ~add~ in the code block below.

2. Scan two integers `100` and `200` as input from an input file.
3. Use the function anywhere inside the body of a `main` function, with the two input arguments, and print the result.

```
// function definition

// main function
```

Solution

```
echo "100 200" > ./src/input
cat ./src/input
```

```
// function definition
int add(int i, int j) { return (i + j); }

// declarations
#include <stdio.h>

// main function
int main(void) {

  // declare variables
  int k, l;

  // get input
  scanf("%d%d", &k, &l);

  // use add() function
  int sum = add(k,l);

  // print result
  printf("%d + %d = %d\n", k, l, sum);

  return 0;
}
```

```
100 + 200 = 300
```

Footnotes:

[1] If you want to create a code block and tangle it, you need to add the header arguments `:main no` :includes none~.

Created: 2022-06-20 Mon 23:11