C functions practice

Table of Contents

README

- Practice workbook for functions in C
- See GitHub for script/solutions

Emacs setup (optional)

Hide emphatic characters like ~, *

To **not** see the emphatic characters like ~ or * or / in the Org file text, run the following code chunk (or put the code in your `/.emacs` file): if successful, you should see `"t"` in the minibuffer.

```
(setq-default org-hide-emphasis-markers t)
```

This will only work for new buffers. If you don't put it in your `/.emacs` file, the command will only work for the current Emacs session.

Close and reopen this file to see an effect.

Change your theme

- In Emacs, type `M-x custom-themes`
- In the buffer that appears, select `Leuven`
- Select `Apply and Save Setting`
- This will work immediately

Recap: `hello_world` function

- `[ ]` The `hello_world` function does not have a `return` value, and it takes no arguments. Fix the code so that it compiles and runs.

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

```
// function call
hello_world();
```

```
Hello world
```

Save a function return value

- [ ]

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

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

  ```
  // 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("Average of %g and %g: %g\n", x, y, avg);
  ```

  ```
  Average of 5.1 and 8.9: 7
  ```

- *Tip: remember to declare your variable. Use `%g` for the output.*

Write a simple function

- [ ]

  Write a function `add` that adds two integer numbers `num1` and `num2`. In the same code block, call the function for the values `2` and `2`, and print the `return` value.

  ```
  // function definition
  int add (int num1, int num2)
  {
    return num1 + num2;
  }

  // function call and print
  printf("2 + 2 = %d \n",  add(2,2));
  ```

  ```
  2 + 2 = 4
  ```

Use a simple function

- [ ]

Put the definition of `add` in the code block below, then use the function inside a `main` function body.

```
// function definition
int add (int num1,int num2){return num1 + num2;}

// main function
int main(void) {

  // function call and print
  int i = 2, j = 2;
  printf("%d + %d = %d \n", i, j, add(i,j));

  return 0;
}
```

```
2 + 2 = 4
```

With arbitrary integer input:

Input:

```
echo 2 2 > add
```

Code:

```
// function definition
int add (int num1,int num2){return num1 + num2;}

// main function
int main(void) {

  int i, j;

  // get input
  printf("Enter two integer numbers: \n");
  scanf("%d %d", &i, &j);

  // function call and print
  printf("%d + %d = %d \n", i, j, add(i,j));

  return 0;
}
```

```
Enter two integer numbers:
2 + 2 = 4
```

Created: 2022-04-22 Fri 10:23