CODING CONVENTIONS

I Naming variables

- Style:
  - Camel case style with the first letter in lowercase: `totalMoney`
  - All letters in lowercase with underscore (_): `total_money`
- Choose meaningful names.
  - For example, name the variable used to store the total money
  - Should: `totalMoney`, `total_money`
  - Should not: ~~temp1~~, ~~abc~~,...

II Naming functions

- Style:
  - Camel case style with the first letter (of function's name) in lowercase: `calculateTotalMoney(...)`
  - Camel case style with the first letters of words (in function's name) in uppercase: `CalculateTotalMoney(...)`
- Name the function with a verb as the first word in the name.
  - Ex: The function calculates the total money
  - Should: `CalculateTotalMoney(...)`
  - Should not: ~~TotalMoney(...)~~, ~~Money(...)~~,...

III Comments

- Line comment:

```
// This is a comment line
```

- Block comment:

```
/*
   This is a comment block
   Line 1
   Line 2
   ...
*/
```

IV Open/close code blocks

- Style 1:

```
if (is_student == true) {
   // Do something
} else {
   // Do something else
}
```

- Style 2:

```
if (is_student == true)
{
    // Do something
}
else
{
    // Do something else
}
```

V Indents and white spaces

- Rule 1: Child code blocks must be indented 1 tab or 4 spaces from the parent blocks.
  - Should:

```
int i;
int sum = 0;

for (i = 0; i <= 100; i++)
{
    cout << i << "\n";
    sum += i;
}
```

  - Should not:

```
int i;
int sum = 0;

for (i = 0; i <= 100; i++)
{
cout << i << "\n";
        sum += i;
}
```

- Rule 2: Operators and operands must be separated by a space.
  - Should:

```
int first_number;
int second_number;
int sum_of_two_numbers;

first_number = 5;
sencond_number = 10;

sum_of_two_numbers = first_number + second_number;
```

  - Should not:

```
int first_number;
int second_number;
int sum_of_two_numbers;

first_number=5;
sencond_number=10;

sum_of_two_numbers=first_number+second_number;
```

- Rule 3: Statement components must be separated by spaces, with punctuation marks (semicolon (;), comma (,), colon (:), etc.) placed close to the preceding component.
  - Should:

```
int i;

for (i = 5; i <= 50; i++)
{
    // Do something
}
```

  - Should not:

```
int i;

for (i=5;i<=50 ;i++)
{
    // Do something
}
```

- Rule 4: Group "related" lines of code together and separate them from "unrelated" lines of code with 1-2 blank lines. Add comments to each code block.
  - Should:

```
// Declare variable
int first_number;
int second_number;
int sum_of_two_numbers;

// Assign data to the variable
first_number = 5;
sencond_number = 10;

// Calculate the sum of two numbers and print it
sum_of_two_numbers = first_number + second_number;
cout << first_number << " + " << second_number << " = " << sum_of_two_numbers;
```

  - Should not:

```
int first_number;
int second_number;
int sum_of_two_numbers;
first_number = 5;
sencond_number = 10;
sum_of_two_numbers = first_number + second_number;
cout << first_number << " + " << second_number << " = " << sum_of_two_numbers;
```

The end