

SIT102 Introduction to Programming

Pass Task 2.1: Hello User

Overview

Now that we have variables, constants, and functions we can create a programs that use these to perform interesting tasks. Get started by reading in the users name from the Terminal and echoes back a welcome message, along with some important calculations. Learn how to work with variables, constants, and functions.

Submission Details

Use the instructions on the following pages to create your own Hello User program.

Submit the following files to OnTrack.

- Your program code
- A screen shot of your program running
- Answers to the supplied questions

The focus of this task is on:

- Declaring and using local variables
- Declaring and using global constants
- Create and calling functions
- Passing parameters
- Declaring and using constants

Your Task

Follow the steps in the week 2 videos to build a program with the following features. The relevant sections of the code are shown after the instructions.

- Ask the user to enter their name and age
- Calculate and output the air speed velocity of the following birds:
 - African Swallow: frequency 15hz, amplitude 21cm
 - European Swallow: frequency 14hz, amplitude 22cm
 - Another bird, where the user enters the frequency and amplitude and name for the bird.

For details on this calculation see [The Strouhal Number in Cruising Flight](#) and [Estimating the Airspeed Velocity of an Unladen Swallow](#).

- Then use what you learn to create your own function that calculates something based on values the user inputs.

This must demonstrate the creation and use of:

- Local variables for the name, age, and others as needed.
- Global constant for the Strouhal number
- Functions for the following:
 - A function of your **own creation**. For example, calculate an AFL score from points and goals, circle area given a radius, BMI given weight and height, or any other value where you can show the use of parameters and returning a value from a function.

Have the user input the necessary details, and then output the result of your calculation.

- To calculate the air speed velocity, with parameters for frequency and amplitude

```
double air_speed(int frequency, double amplitude)
```

- To read in a string from the user, with a parameter for the message to prompt them with.

```
string read_string(string prompt)
```

Have a look at the [How functions are used video](#) for the instructions needed to read in a string (see how we do this to read in the user's name). Also watch the [How functions are created video](#) for how to code this function.

- To read in an integer from the user, with a parameter for the message to prompt them with.

```
int read_integer(string prompt)
```

Have a look at the [How functions are used video](#) for the instructions needed to read in an integer (see how we do this to read in the user's age). Follow the same steps used to create the read string function in [How functions are created video](#) to create the read integer function.

- To read in a double value from the user, with a parameter for the message to prompt them with.

```
double read_double(string prompt)
```

This will be very similar to **read integer**, in this case you can **convert_to_double** rather than converting to an integer.

Make sure to adjust **all** user input to use your new **read_string**, **read_double**, and **read_integer** functions.

Reminders

- Use the following instructions for setting up your project in the terminal.

```
cd /c/Users/andrew/Documents/Code
mkdir HelloUser
cd HelloUser
skm new c++
```

- Open **Visual Studio Code** and open the *folder* you created.

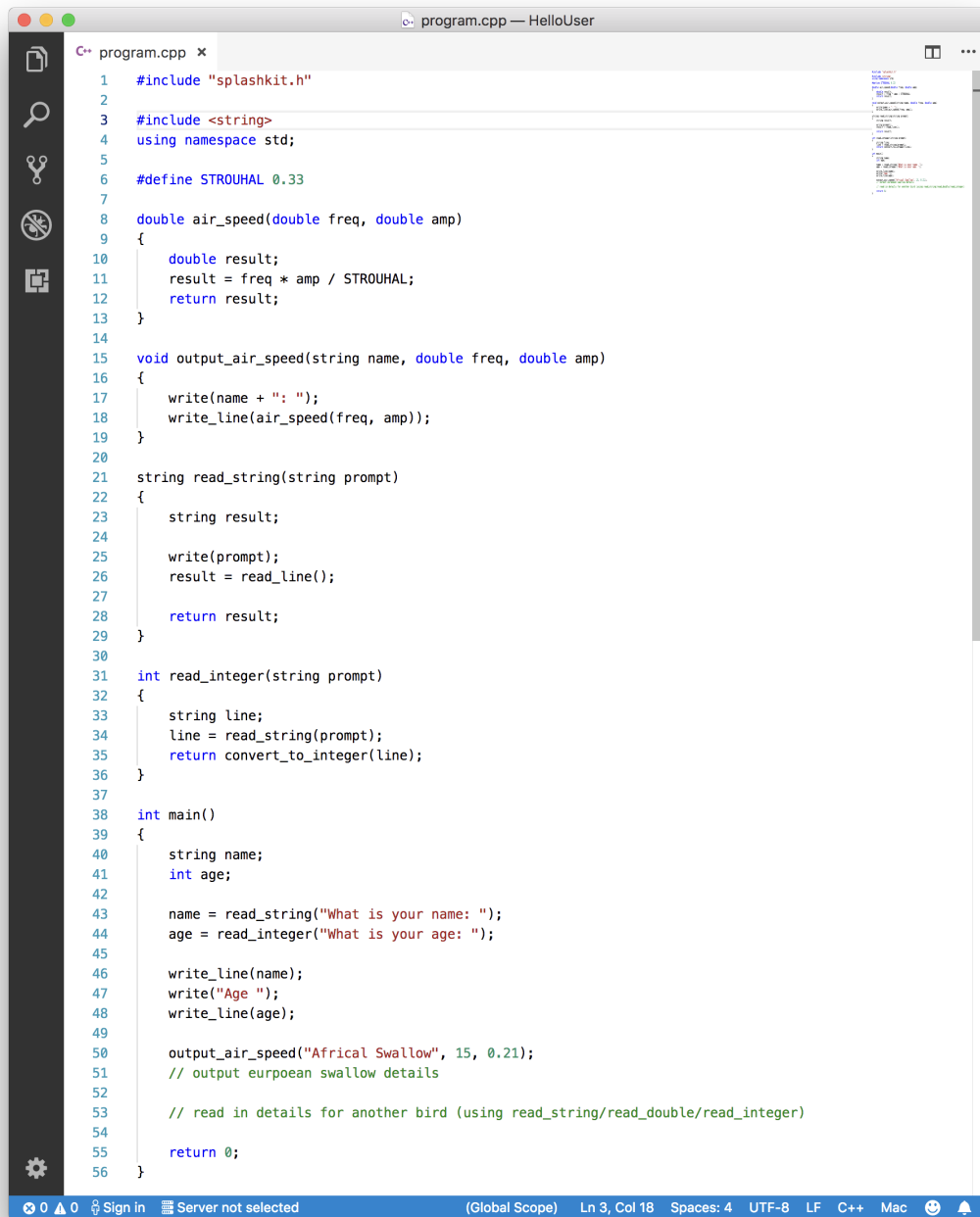
Remember to make sure to open the **folder** (directory) not just the file. This will let VS Code find all of the files related to your project, including the SplashKit files.

- Compile and run the program using the following instructions:

```
skm clang++ program.cpp -o HelloUser
./HelloUser
```

- Splashkit includes a number of functions that can help:

Function	Task
<code>void write(string message);</code>	Writes a message to the terminal, and stay on the same line.
<code>void write_line(string message);</code>	Writes a message to the terminal, and goes to the next line.
<code>string read_line();</code>	Reads a line of text from the user and returns it.
<code>int convert_to_integer(string text);</code>	Convert the text into an integer, and return the value read.
<code>double convert_to_double(string text);</code>	Convert the text into a double, and return the value read.



```
1  #include "splashkit.h"
2
3  #include <string>
4  using namespace std;
5
6  #define STROUHAL 0.33
7
8  double air_speed(double freq, double amp)
9  {
10     double result;
11     result = freq * amp / STROUHAL;
12     return result;
13 }
14
15 void output_air_speed(string name, double freq, double amp)
16 {
17     write(name + ": ");
18     write_line(air_speed(freq, amp));
19 }
20
21 string read_string(string prompt)
22 {
23     string result;
24
25     write(prompt);
26     result = read_line();
27
28     return result;
29 }
30
31 int read_integer(string prompt)
32 {
33     string line;
34     line = read_string(prompt);
35     return convert_to_integer(line);
36 }
37
38 int main()
39 {
40     string name;
41     int age;
42
43     name = read_string("What is your name: ");
44     age = read_integer("What is your age: ");
45
46     write_line(name);
47     write("Age ");
48     write_line(age);
49
50     output_air_speed("Africal Swallow", 15, 0.21);
51     // output eurpoean swallow details
52
53     // read in details for another bird (using read_string/read_double/read_integer)
54
55     return 0;
56 }
```

0 0 Sign in Server not selected (Global Scope) Ln 3, Col 18 Spaces: 4 UTF-8 LF C++ Mac

Figure: Code from the videos for the program

Step 3: Submit your work

Do the following before you submit:

- Check to make sure you have the following:
 - Functions to read user input: **read_string**, **read_integer**, and **read_double**.
 - A function to calculate **air speed**.
 - Calls to test different bird values
 - Your *own* function to calculate some value
- Download the task resources, and answer the questions in the Word file. Save as PDF to prepare for upload.
- Run your program and grab a screenshot of your scene.

When you are ready, login to [OnTrack](#) and submit your code, screenshot, and answers to Pass Task 2.1.

Remember to save and backup your work! Storing your work in multiple locations will help ensure that you do not lose anything if one of your computers fails, or you lose a USB Key.

Task Discussion

Discuss the following with your tutor to demonstrate your understanding of the concepts covered.

- Explain the difference between a variable and a constant. When would you use each of these?
- Why use constants when the value could just be typed in the code everywhere it is needed?
- What are some of the different types you can use for the values in your variables?