Hands-on Activity 6.2	
Built-in function	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed: 29/10/25**
Section: CPE11S1	**Date Submitted: 30/10/25**
Name(s): James Daniel M. Verano	**Instructor: Engr. Jimlord M. Quejado**
6. Output	

⇒ Activity 1:

Code:

```
1   #include <iostream>
2   using namespace std;
3
4   float cube (float S);
5
6   int main() {
7
8       int S;
9       float result;
10
11      cout << "Input cube's length: ";
12      cin >> S;
13      result = cube(S);
14      cout << "The Cube's value is: " << result;
15
16
17      return 0;
18  }
19
20  float cube (float S) {
21      float convert = S * S * S;
22      return convert;
23  }
```

Output/s:

```
Input cube's length: 25
The Cube's value is: 15625
--------------------------------
Process exited after 8.474 seconds with return value 0
Press any key to continue . . .
```

⇒ [ANALYSIS]:

In this first activity example, I have created a program that defines a function to compute for the volume of cube with a formula of V = s * s * s. line code 2 - 4, are our function declarations for our necessities for our computation for the volume. Now, after the declaration of functions for our variables to use in our computation, we will now define the computation's initialization at the outer bottom part of the main block. We will create our function definition there. After that, in the main block function, we will be creating a user input to get the needed values in order for the computation to start looking for its cube's volume value. Lastly, define the function in the main block, and print out the result.

⇒ Activity 2:

Code:

```cpp
#include <iostream>
#include <cmath>
using namespace std;

double hypotenuse (double a, double b);

int main() {

    double a, b, result;

    cout << "Enter 1st length value: ";
    cin >> a;
    cout << "Enter 2nd length value: ";
    cin >> b;

    result = hypotenuse(a,b);

    cout << "The value of the hypotenuse is: " << result;

    return 0;
}

double hypotenuse (double a, double b) {
    double convert = sqrt ((pow (a, 2) + pow (b, 2)));
    return convert;
}
```

output/s:

```
Enter 1st length value: 100
Enter 2nd length value: 100
The value of the hypotenuse is: 141.421
--------------------------------
Process exited after 3.876 seconds with return value 0
Press any key to continue . . .
```

⇒ [ANALYSIS]:

In this second activity example, I have created the objective's intended program, which is to define a function hypotenuse that calculates the length of the hypotenuse of a right triangle when the other two sides are given. In my program, you can see that in the outer upper of the main block, is a defined function for the hypotenuse calculation. After the function declaration, we can proceed at the outer bottom of the main block function. You will encounter the function definition for the computation of the hypotenuse. I used the sqrt***()*** and ***pow()*** function as it is needed for the formula of finding the value of the hypotenuse. After the function definition, Same as in the first activity example, in the main block section, we will create a user input for the two sides given. And once the user inputs their given numbers, the defined function below will be initiated and we will be defining the function definition's return function for the result. And just print out the result.

⇒ Activity 3:

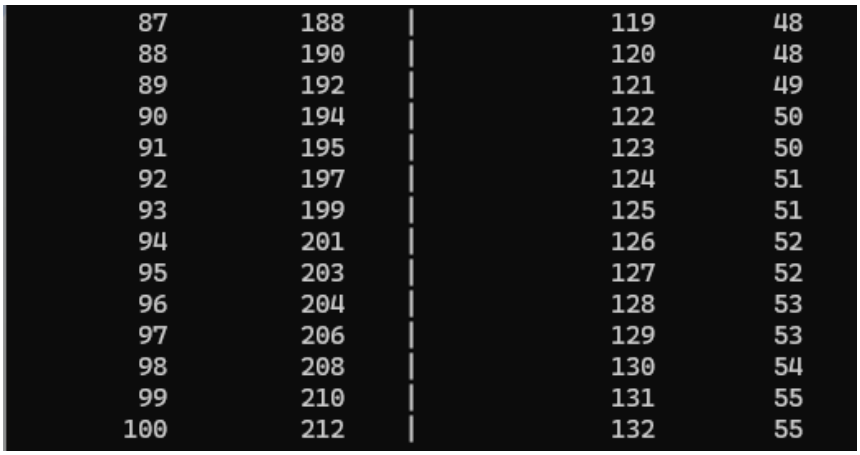
Code:

```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int celsius(int f) {
    return static_cast<int>((f - 32) * 5.0 / 9);
}

int fahrenheit(int c) {
    return static_cast<int>(c * 9.0 / 5 + 32);
}

int main() {

    cout << setw(10) << "Celsius" << setw(12) << "Fahrenheit"
    << "     |     " << setw(12) << "Fahrenheit" << setw(10) << "Celsius" << endl;
    cout << string(55, '-') << endl;

    for (int c = 0, f = 32; c <= 100 && f <= 212; c ++, f ++) {
        cout << setw(10) << c << setw(12) << fahrenheit(c)
        << "     |     "
        << setw(12) << f << setw(10) << celsius(f) << endl;
        }
    return 0;
}
```

Output/s:

```
   Celsius  Fahrenheit     |      Fahrenheit   Celsius
-------------------------------------------------------
         0          32     |           32         0
         1          33     |           33         0
         2          35     |           34         1
         3          37     |           35         1
         4          39     |           36         2
         5          41     |           37         2
         6          42     |           38         3
         7          44     |           39         3
         8          46     |           40         4
         9          48     |           41         5
        10          50     |           42         5
        11          51     |           43         6
        12          53     |           44         6
        13          55     |           45         7
        14          57     |           46         7
        15          59     |           47         8
        16          60     |           48         8
        17          62     |           49         9
        18          64     |           50        10
        19          66     |           51        10
        20          68     |           52        11
        21          69     |           53        11
        22          71     |           54        12
        23          73     |           55        12
        24          75     |           56        13
        25          77     |           57        13
        26          78     |           58        14
        27          80     |           59        15
```

28	82	60	15
29	84	61	16
30	86	62	16
31	87	63	17
32	89	64	17
33	91	65	18
34	93	66	18
35	95	67	19
36	96	68	20
37	98	69	20
38	100	70	21
39	102	71	21
40	104	72	22
41	105	73	22
42	107	74	23
43	109	75	23
44	111	76	24
45	113	77	25
46	114	78	25
47	116	79	26
48	118	80	26
49	120	81	27
50	122	82	27
51	123	83	28
52	125	84	28
53	127	85	29
54	129	86	30
55	131	87	30
56	132	88	31
57	134	89	31
58	136	90	32
59	138	91	32
60	140	92	33
61	141	93	33
62	143	94	34
63	145	95	35
64	147	96	35
65	149	97	36
66	150	98	36
67	152	99	37
68	154	100	37
69	156	101	38
70	158	102	38
71	159	103	39
72	161	104	40
73	163	105	40
74	165	106	41
75	167	107	41
76	168	108	42
77	170	109	42
78	172	110	43
79	174	111	43
80	176	112	44
81	177	113	45
82	179	114	45
83	181	115	46
84	183	116	46
85	185	117	47
86	186	118	47
87	188	119	48



87	188	119	48
88	190	120	48
89	192	121	49
90	194	122	50
91	195	123	50
92	197	124	51
93	199	125	51
94	201	126	52
95	203	127	52
96	204	128	53
97	206	129	53
98	208	130	54
99	210	131	55
100	212	132	55

⇒ [ANALYSIS]:
In this final activity example, I have created the intended program objective, which is to perform a temperature conversion of Celsius to Fahrenheit and vice versa. So in this example, after performing and declaring a conversion, we are going to use those functions to also write a program that prints charts showing their equivalents. In terms of Fahrenheit equivalents of a celsius temperature from 0 to 100 degrees, and for the vice versa of equivalents, it will be from temperature 32 to 212 degrees.

After defining these functions, we will proceed now to the main function, which is where the display and formatting of the conversion table. We also used the library **<*Iomanip*>** for a more organized table structure for the addition function setw() for spacing. We used a for loop to iterate through the range of all the temperature values in terms of their equivalents, which was the 0 - 100 C, and 32 - 212 F. Once the loop condition is met, then the table will show the temperatures that the activity intends to achieve.

7. Supplementary Activity

- Screenshot of Code:
- Output of Code (label and compile ALL possible outputs):
- Short Analysis(min 5 sentences):

8. Conclusion

In conclusion, throughout the analysis, I was able to beautifully execute and define functions in a clean and neat way to represent conversions, declare variables, and initialize functions. I was also quite introduced into using new functions and was guided to a more advanced way to create a formula using the function I was introduced to. These programs have deeply enhanced and bound my experience in creating a program using functions for a more formal and cleaner approach in terms of declaring variables and defining functions. Although, my analysis wasn't explained well due to the lack of time I have, yet I am still trying to look for improvements and visuals for possible learnings to explore new potentials and skills.