EUROPEAN UNIVERSITY OF LEFKE
Faculty of Engineering
Department of Computer Engineering

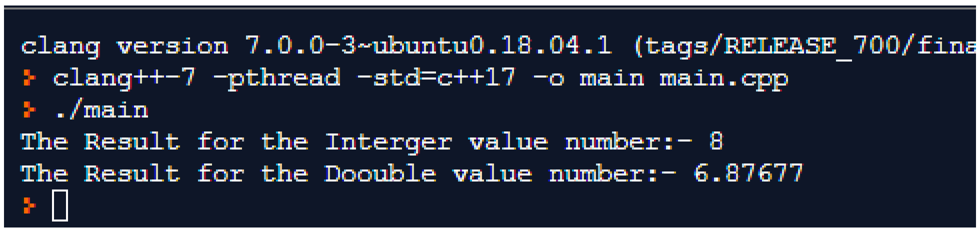
COMP218
OBJECT-ORIENTED PROGRAMMING

LAB WORK NO. 3

Prepared by **David O. Ladipo** (174574)
Submitted to Dr. Ferhun Yorgancıoğlu

Task-1: Write a C++ program that gives the definition of two overloaded functions for calculating the distance between two discrete points in the Cartesian coordinate system. First variant takes into consideration of integers only. Second variant extends it double values. Pay attention to the fact that both functions shall be named same but declared with different signatures. The distance formula: $d = \sqrt{}(y2 - y1)2 + (x2 - x1)2$.
Note: Use a template function definition to minimize both functions into one instead of having two.

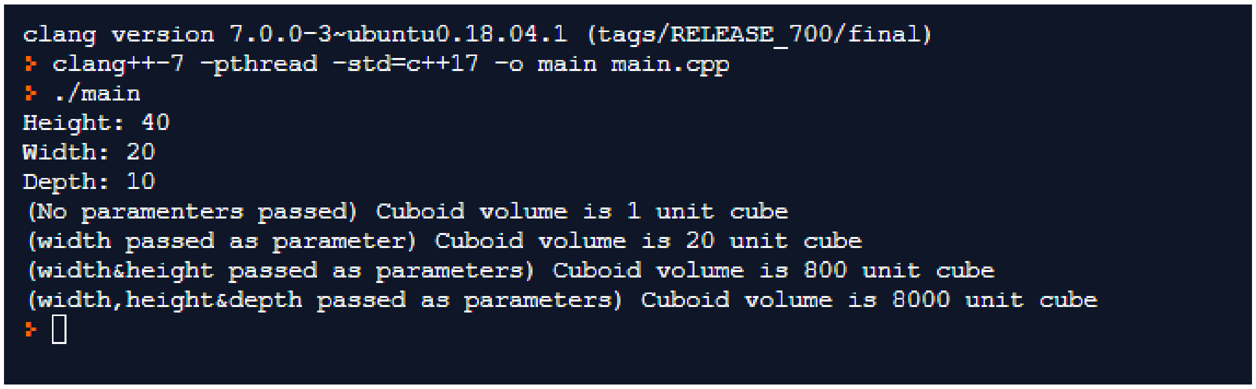
```
#include <iostream>
#include <cmath>
using namespace std;

//Template function created to minimize both functions into one
template<class T>
inline T distance(T y1,T y2,T x1,T x2){
  return sqrt(  pow(y2 - y1,2) + pow(x2 - x1,2)  );
}

// inline function to calculate distance of integer input
 inline int distance(int y1,int y2,int x1,int x2){
   return sqrt(  pow(y2 - y1,2) + pow(x2 - x1,2)  );
 }
//inline function to calculate distance of double input
 inline double distance(double y1,double y2,double x1,double x2){
   return sqrt(  pow(y2 - y1,2) + pow(x2 - x1,2)  );
 }

int main() {
  //declared integer values for the Cartesian coordinates
  int x1=10, x2=12, y1=7, y2=15, integer_Result;
  //declared double values for the Cartesian coordinates
  double x_1=1.0, x_2=5.5, y_1=5.0, y_2=10.2, double_Result;

  //integer function for distance is called and result stored in integer_Result
  integer_Result = distance(y1,y2,x1,x2);
  //Double function for distance is called and result stored in double_Result
  double_Result = distance(y_1,y_2,x_1,x_2);

  cout<<"The Result for the Interger value number:- "<<integer_Result<<endl;
  cout<<"The Result for the Doouble value number:- "<<double_Result<<endl;
}
```



```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/fina
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
The Result for the Interger value number:- 8
The Result for the Doouble value number:- 6.87677
>
```

Task-2: Write a C++ program that uses *default arguments* in a function for calculating the volume of a cuboid. All dimensions, namely, the width, the height and the depth shall be defaulted to 1 unit, respectively. Test your program with different function calls!
Note: Provide both function prototype and definition separately to observe the usage of default values. Can you repeat default values in the definition?

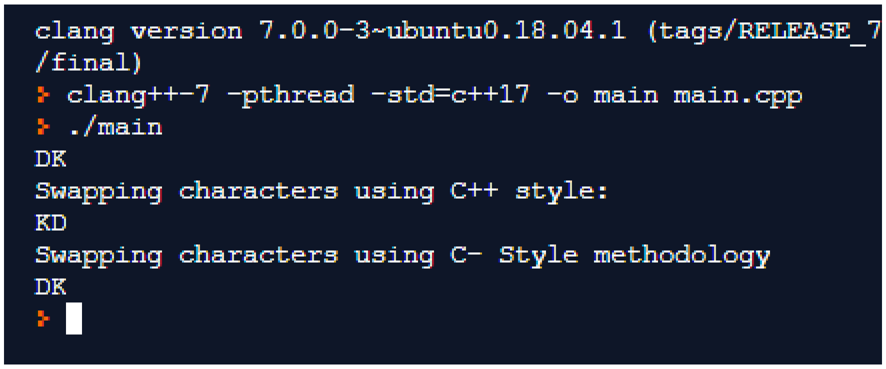
```
#include <iostream>
using namespace std;

// function to calculate the volume of a cuboid and default arguments set to 1
respectively, if width, height or depth is not given, it is assumed to be 1.
int volume_of_a_cuboid(int width=1,int height = 1,int depth = 1){
return width * height * depth;
}

int main() {
int height,width,depth,cuboid_volume; // variable declared respectively
cout<<"Height: ";
cin>>height; //stores input value of height

cout<<"Width: ";
cin>>width; //stores input value of width

cout<<"Depth: ";
cin>>depth; //stores input value of depth

cuboid_volume = volume_of_a_cuboid(); // function call with no parameters

cout<<"(No paramenters passed) Cuboid volume is "<<cuboid_volume<<" unit
cube"<<endl;

cuboid_volume = volume_of_a_cuboid(width); // function call with width
cout<<"(width passed as parameter) Cuboid volume is "<<cuboid_volume<<" unit
cube"<<endl;

cuboid_volume =volume_of_a_cuboid(width,height);//function call with width&height
cout<<"(width&height passed as parameters) Cuboid volume is "<<cuboid_volume<<"
unit cube"<<endl;

uboid_volume = volume_of_a_cuboid(width,height,depth);//function call with
width,heigh&depth
cout<<"(width,height&depth passed as parameters) Cuboid volume is
"<<cuboid_volume<<" unit cube"<<endl;
}
```



```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final)
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
Height: 40
Width: 20
Depth: 10
(No paramenters passed) Cuboid volume is 1 unit cube
(width passed as parameter) Cuboid volume is 20 unit cube
(width&height passed as parameters) Cuboid volume is 800 unit cube
(width,height&depth passed as parameters) Cuboid volume is 8000 unit cube
> 
```

Task-3: Write a C++ program that uses *reference parameters* in a function for exchanging the values between two characters. Test your program with different set of arguments in the function calls. Provide another version of the function for doing the swapping but this time using pointers to achieve C-style call-by-reference methodology. Observe the difference between the strategies!
Note: Can reference parameters be defaulted as well?

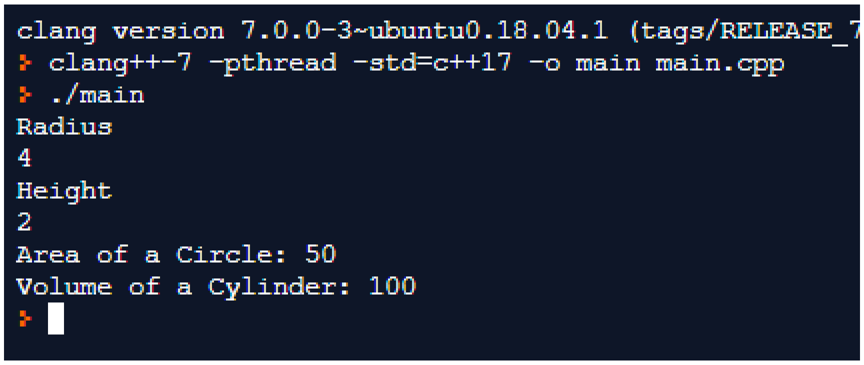
```
#include <iostream>
using namespace std;

//C-style call by reference methhodology
inline void swap(char *a,char *b){
char temp = *a;
*a = *b;
*b = temp;
}

//C++ call by reference
inline void swap(char &a,char &b){
char temp = a;
a = b;
b = temp;
}

int main() {
char firstChar='D',secondChar='K';
cout<< firstChar << secondChar <<endl;
cout << "Swapping character susing C++ style: " << endl;
swap(firstChar,secondChar);
cout <<firstChar <<secondChar << endl; // swapping was done effectively because
c++ call by refeence method will have effect directly on the values in the main
function.
cout << "Swapping characters using C- Style methodology" << endl;
swap(&firstChar,&secondChar);
cout<<firstChar<<secondChar<<endl; // Swapping was not done using c-style method
because the function has no effect on the values in the main.
}
```



```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_7
/final)
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
DK
Swapping characters using C++ style:
KD
Swapping characters using C- Style methodology
DK
>
```

Task-4: Write a C++ program that gives definitions of *inline functions* for calculating the area of a circle and the volume of a cylinder. The area of a circle can be calculated using $\pi r2$ whereas the volume of a cylinder with $\pi r2h$. Pay attention to the fact that the second function's definition might get the benefit of the first one. Test your program with different set of arguments in the function calls.
Note: A macro can be written to achieve the same effect! Try it!

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

//Template function for finding both Area of a circle and Volume of a Cylinder
with h sets to default 1
template<class D>
inline D areaVolume(D r,D h=1){
  float pi=3.142f;
  return pi*r*r*h;
}

int main() {
  int radius,height,circle,cylinder;
  cout << "Radius\n";
  cin>>radius; //saves the user input to radius
  cout << "Height\n";
  cin>>height; //saves the user input to height
  circle = areaVolume(radius);// calls the areaVolume function,pases radius as a
parameter to calculate the Area of the Circle
  cout << "Area of a Circle: "<<circle<<endl;
  cylinder = areaVolume(radius,height);// calls the areaVolume function,pases
radius and height as parameters to calculate the Volume of the Cylinder
  cout << "Volume of a Cylinder: "<<cylinder<<endl;
}
```



```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_7
› clang++-7 -pthread -std=c++17 -o main main.cpp
› ./main
Radius
4
Height
2
Area of a Circle: 50
Volume of a Cylinder: 100
›
```