Submission Form

Please ensure you fully complete all FOUR questions on this form before submitting your assignment.

1. Please list the programming constructs you have used.

If statements

Else statements

2. Describe in simple language how your program works.

The program asks the user to enter the length, diameter and actual weight of a piece of fibreglass. It then outputs those values back to the user. We then create the prototype for the isValidDimensions function, this is then used in an IF statement with length and diameter as arguments set to return a message about the dimensions not being valid in the instance where isValidDimensions is NOT true.

The isValidDimensions function checks using an IF statement to ensure that the user entered length is between 1-20mm AND the diameter is between 0.1 -1mm, if this is the case then it returns true - if it is not the case, then an else statement is used and nothing is returned, which in turn means that the condition of the main program's IF statement is met and a message about dimensions not being valid is sent to display.

In the instance that isValidDimensions is true, then the else statement is used. Within the else firstly we create a prototype function for eval_volume, we then assign the variable est_volume as equal to the value returned from the function eval_volume, which takes the arguments length and diameter.

eval_volume calculates the volume of the piece of fibreglass using the formula for a cylinder, which for our purposes is: Volume = $3.142x(diameter/2)^2xlength$, then returns that value.

Then we do the same for eval_weight, creating a prototype and assigning est_weight to the value of the function. It takes two arguments, est_volume that was returned from our previous function and 0.05 which is the conversion rate of $1mm^3$ to weight in grams.

The function multiplies the estimated volume by the conversion rate and returns it as the calculated estimated weight.

Then finally we create a function prototype for "wedge_or_cylinder". The function takes two arguments in the form of previously calculated estimated weight and the user entered actual weight and calculates if it's a wedge or cylinder.

The function takes our two values, firstly it uses an IF statement to check if actual weight is less than 0 OR if a non-double data type has been entered, if that is the case then it simply displays that the weight is invalid as you can't have negative weight or a non-numeric weight. If this is not the case then the ELSE statement is used. The function then calculates the difference between the two weights, assigning the difference to a variable "diff". This variable is then put into an if statement which checks if diff is greater than 0.1, if so then it's defined as Wedge shaped, and if it's not then it's Cylinder shaped. It then outputs this result to display. The program then ends.

3. Please describe below any additional features that you've included in your programme, if none what would you add if you had more time?

I added the additional feature of outputting the values along with their headings, this was largely a debugging feature so I could check that my calculations were correct, as well as outputting the shape as the brief only specified calculating which of the given two it was.

I have added some input validation to the actual weight input, both diameter and length have strict input restrictions - the same is not true for weight. However due to there being no real restriction on the weight of a given 'chip' I simply have it restricted such that the actual weight cannot be less than 0, as from a purely physical standpoint it's not possible to have something weigh a negative value. I used less than 0, as it's possible due to the fact we're measuring in grams and we don't know the accuracy of the system being used to weigh the object that it may be rounded down to a weight of 0, and I wouldn't want the possibility of somebody entering a value that their scales has and it reading as invalid, even if it technically doesn't weigh "0" necessarily.

I also have it validated such that you cannot enter an incorrect data type, as without that piece of code it would be possible to enter a letter for instance and it would register the actual weight as 0 rather than being treated as invalid which in turn would mean that a shape can be given despite the lack of a valid actual weight.

Beyond that due to the highly specialized nature of the program I don't think there's anything to add that remains within the scope of the pre-written program or is necessary to be added.

4. Please paste your code into the box below, ensuring it has been formatted correctly

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

int main() {
    double length = 0;
    double diameter = 0;
    double act_weight = 0;
    double est_volume = 0;
    double est_weight = 0;

    cout << "Enter the length:" << endl;
    cin >> length;

    cout << "Enter the diameter" << endl;
    cin >> diameter;

    cout << "Enter the actual weight: " << endl;
    cin >> act_weight;

    cout << "Length: " << length << endl;
    cout << "Diameter: " << diameter << endl;
    cout << "Actual Weight: " << act_weight << endl;

    bool isValidDimensions(double, double);
    if (!isValidDimensions(length, diameter))
    {
        cout << "Dimensions are not valid";
    }
    // check len is within the range 1mm to 20mm and dia is in range 0.1mm - 1mm
    else
    {
        //estimate volume
        double eval_volume(double, double);
        est_volume = eval_volume(length, diameter);
        // estimate weight
        double eval_weight(double, double);
        est_weight = eval_weight(est_volume, 0.05);
        // display if wedge or cylinder
        void wedge_or_cylinder(double, double);
        wedge_or_cylinder(est_weight, act_weight);
    }
    return 0;
}


//FUNCTIONS
bool isValidDimensions(double len, double dia) { //ensure that different parameter names are used to
arguments for good practice.
    if (len >= 1 && len <= 20 && dia >= 0.1 && dia <= 1) {
        return true;
    }
    else {
        return 0; // this is not technically necessary in VS, but for other compilers without it an
error about reaching the end of a non-void function may occur so I'll keep it here.
    }
}

double eval_volume(double len, double dia) {
    double estimatedVol = 3.142 * ((dia / 2) * (dia / 2)) * len; //multiplying by each other,
mathematically the same as squaring. Other methods can be used but this requires no additional
libraries or functions.
    return estimatedVol;
}
```

```
double eval_weight(double estimatedVol, double weight) {
    double estimatedWeight = estimatedVol * weight;
    return estimatedWeight;
}

void wedge_or_cylinder(double estimatedWeight, double actualWeight) {
    if (actualWeight < 0 || !cin) { //ensure actualweight is a valid value, not negative or not an
invalid data type.
        cout << "Weight is invalid";
    }
    else {
        double diff = actualWeight > estimatedWeight ? actualWeight - estimatedWeight :
estimatedWeight - actualWeight; //use of the ternary operator - essentially a condition, if act
weight is greater than est weight then it reads as true and the code on the left of the colon will be
used, if false then the code on the right will be used.
        cout << "Shape: ";
        if (diff > 0.1) {
            cout << "Wedge Shaped" << endl;
        }
        else {
            cout << "Cylinder Shaped" << endl;
        }
    }
}
```