C++ code kata: Week 1 part 1 of 3

Undersanding variables and literals

Contents

Hello 👋

If you have not attempted 💻, I would request you to try your hand over few solutions, for those of you who have written soem solutions here is solution document 🎁 for you.

Solutions

01. Age of a Person

```
01:     std::cout << "Age of a person: " << 78 << std::endl;
02:     std::cout << "Age of person: " << std::dec << 78 << std::endl;
03:     printf("Age of person: %u\n", 78);
04:     fprintf(stdout, "Age of person: %u\n", 78);
```

Solution 1: Commentry

In our solutions we have written this statement in four ways let us discuss them one by one:-

line 01

We have used cout, which is object of ostream class and is associated with standard output device 🖹.
Since this class is defined in standard namespace we have to prefix `std::` before using `cout`.

In first technique *see line 01* we have passed a string literal followed by age of person. Since age will always be a numeric value and we are under assumption that it will be integral value we have written age as an integer literal.

Remember integers can be written in **binary, decimal, octal and headecimal number systems** in C/ C++. Age is best expressed in decimal number system that we have used here.

The << used here is known as **insertion operator** is basically an overloaded operator of ostream class to handle various kind of `literals`.

At last the std::endl is a manipulator that inserts a new line character in the stream.

line 02

Same as **line 01**, but the only change here is the std::dec, which is a manipulator that modifies the default numeric base for integer I/O. It is useful when you have to represent an integer entity in a particular base.

This is not quite useful here as we are already in same base.

line 03

This solution is interesting in sense that some of the new budding C++ programmers don't even consider it a C++ solution. C language is subset of C++ and printf is a valid C++ function for C style I/O. It takes a format string with specifier that is required for output conversion.

Since it is a function, the arguments are passed visibly (unlike Object orientation), and each is separated by a `comma (,)` operator.

Out of all `format specifiers` for integers the `%u` is most appripriate here ☹.

line 04

Same as **line 03**, but the only change here is fprintf again a valid C++ function for C style I/O. It takes an additional argument stream representing the FILE object, basically representing where the output will go.

02. Value of absolute zero in celcius

```
01:     std::cout << "Value of absolute zero: " << -273.15F << std::endl;
02:     std::cout << "Value of absolute zero: " << std::fixed << -273.15F << '\n'
03:               << "Value of absolute zero: " << std::scientific << -273.15F <<
'\n'
04:               << "Value of absolute zero: " << std::hexfloat << -273.15F <<
'\n'
```

```
05:              << "Value of absolute zero: " << std::defaultfloat << -273.15F
<< '\n';
06:     printf("Value of absolute zero: %f\n", -273.15F);
07:     printf("Value of absolute zero: %e\n", -273.15F);
08:     printf("Value of absolute zero: %g\n", -273.15F);
```

Solution 2: Commentry

In our solutions we have written this statement in four ways let us discuss them one by one:-

line 01

Value of absolute zero in celcius is a fractional number, that can be best represented as a floating point number.

Floating point notation is how fractional numbers are represented in computer 💻. More about it you can read here.

Now we have written value for absolute zero as `-273.15F`. Apart from integers, floating point values are second type of literals used in computer programming. You must be wondering why is that `F` suffixed to value.

It is basically to round off the value to single precision. We will discuss more on this when we are discussing the datatype, variables and memory.

line 02 to 05

Same as **line 01**, but the only change here is the std::fixed and three more manipulators, prefixed and they modify the default formatting for floating-point input/output. It is useful when you have to represent an integer entity in a particular base.

This is not quite useful here but we can put these to use when dealing with irrational numbers like value of mathematical constant likePi.

line 06 to 08

Here we have used printf and have provided various `format specifier` controlling the value conversion for output stream. Lets see when we need them

- %f is used to display decimal point in lowercase.
- %e is used for scientific notation (mantissa/exponent), lowercase.
- %g will choose either of former two whichever is appropriate.

03. Value of PI

```
01:     std::cout << M_PI;
```

Solution 3: Commentry

Let us discuss our solution:-

line 01

We may be thinking to do some mathematical computation like dividing 22 by 7 times or taking value as `3.14`, but both of these approaches are wrong since they depend upon the person writing the code and if some one else makes use of it that person has to make sure that value of PI is correct.

Instead of this the best way is to ask language itself for the value of Pi. To do that you have to define constant `_USE_MATH_DEFINES` and include `cmath` header. `cmath` defines Pi as `M_PI` that we have used in our solution.

> Hope you have liked the solutions and must have learnt some thing. Rest of the answers will be available to you in part 2 and 3 of this answer sheet.

> If you still have questions fell fee to connect. We will be happy to answer your questions.
