Programming review questions

This collection of programming questions includes one question per topic, although some topics do not directly lend themselves towards a programming question. It is meant to gage your strength of understanding of one aspect per topic, to help guide you as to what you need to study. The questions are very targeted towards each topic; however, midterm questions will almost certainly cover numerous topics per question.

1. Write a program that prints “Hello Bob!” to the console output.

2. Why will the following program not compile?

```
int main();

int main() {
    std::cout << "Hello world!" << std::endl;
    return 0;
}
```

What is the purpose of the missing statement or statements?

3. Write a regular expression that describes phone numbers when written as, for example,

+1 519 888 4567

4. What is the regular expression that describes identifiers?

5. The polynomial $x+\frac{1}{3}x^3+\frac{2}{15}x^5+\frac{17}{315}x^7$ approximates the tangent function on the interval [–0.5, 0.5] with an error less than 0.00005. Write an arithmetic expression that evaluates this polynomial with no more than six multiplications.

Bonus: How can you reduce this to four multiplications if you use a local variable?

6. Which of the following produce output, and which produce compile-time errors, given that there is an operation `m << n` where `m` and `n` are integers?

```
((std::cout << 1) << 2) << std::endl;
(std::cout << (1 << 2)) << std::endl;
std::cout << ((1 << 2) << std::endl);
std::cout << (1 << (2 << std::endl));
```

7. Write a function that returns the polynomial in Question 5. Give it an appropriate identifier.

8. Comment your function in Question 7 so that any programmer can understand why the program does what it does.

Note: The approximation to the tangent function is a 7th-order Taylor series.

9. Correct the following code to remove all compile-time errors with code that expresses what the author likely meant:

```
#include iostream

int main()
double f()

int main() {
    cout << "f(3.2) = " << f(3.2) < endl;

    return;
}

double f( double x ) {
    return (xx*(x - 3) + 4)x + 5;
}
```

10. You have been asked to implement a collection of functions that calculate the square root and the n^{th} root of a floating-point number. Write a code skeleton together with appropriate names for the functions together with appropriate function declarations and skeletal function definitions.

11. Explain why decision making processes within a computer must be restricted to a Boolean-valued decision: either a condition is true or it is false.

12. Write a conditional statement that returns true if $b^2 > 4ac$ using each of the four conditional operators `>`, `>=`, `<` and `<=`.

13. Why is the following code sub-optimal?

```
if ( x >= 0 ) {
    return x*x*x;
} else if ( x < 0 ) {
    return -x*x*x;
}
```

14. Write a logical statement that returns true if:

a. $|x| < 10$,
b. $|x| \leq 1$,
c. $0 < x < 1$,
d. $|x| \neq 1$, and
e. the pair (x, y) lies inside the unit circle on the xy-plane but does not equal the point (0, 0).

15. Simplify the following function using a local variable so that fewer operations are performed.

```
double taylor_sin( double x ) {
    return x*(1-(x*x*(1-x*x*(1-x*x*(1-x*x*(1-x*x/110)/72)/42)/20))/6);
}
```

16. Explain why the following code does not change the value of the local variable `a`.

```
void f( int x );
int main();

void f( int x ) {
    x = 3;
}

int main() {
    int a{0};
    f( a );
    return 0;
}
```

17. Write a function that asks the user for a double and then prints out a table with the sine, cosine, tangent, secant, cosecant and cotangent functions evaluated at that point. Be sure to refer to http://www.cplusplus.com/reference/cmath/ to see which functions are available through the standard C math library.

18. Describe the repetitious operation of a traffic light that changes every minute. Any time a light changes, there is a 3 second pause.

19. First, write a function sinc(x) that returns 1 if $x = 0$ and, using the standard library sine function, returns $\sin(\pi x)/(\pi x)$ otherwise. Next, write a function `exceeds_0_5_sinc( double a, double b )` that determines if the sinc function exceeds the value 0.5 on the interval $[a, b]$ on the interval $[a, b]$ that proceeds as follows: initialize a local variable x with the value of a, then while $x \leq b$, test if sinc(x) > 0.5, in which case return true. If no point is found that exceeds 0.5, return false.

20. In your own words, describe why restricting code to conditional statements and while loops is not only sufficient, but also more beneficial than jumping around using the `goto` statement.

21. Write a function that takes an argument r and if $|r| \geq 1$, return 0.0; otherwise, calculates the geometric sum $1 + r + r^2 + r^3 + r^4 + \cdots$ and continues to do so until $|r^n| < 10^{-10}$. In your main function, you should choose a value of r and the print out the result of your function together with the formula $1/(1 - r)$. You should either use `std::abs` in the standard C math library, or write your own absolute value function.

22. Write a function that calculates $n!$. Next, write a function that that takes two arguments, a and n and calculates $a(a + 1)(a + 2)\cdots(a + n - 1)$.

23. Rewrite the game of hi-lo as follows:

a. Ask the user if that individual wishes to play. If no, finish looping and exit.
b. Pick a random number using rand() % 100.
c. Do the following while the individual did not guess the correct number:
    a. Ask for a guess.
    b. If the individual enters the same guess as the previous guess, print "I guess you're bored, we'll try again." and break out of the game.
    c. Otherwise,
        i. if the guess is too high, write "high",
        ii. if the guess is too low, write "low",
        iii. otherwise write "you're right!".
d. Return to Step a.

24. Print the numbers from 0 to 257 in hex using the command `std::cout << std::hex << n;`

Your output should be a comma-separated list, but do not have a leading comma or a trailing comma.

25. Why do these for loops not terminate?

```
for ( unsigned short k{0}; k < 65536; ++k ) {
    // Do something...
}

for ( unsigned short k{65535}; k >= 0; --k ) {
    // Do something...
}
```