Uttara InfoSolutions

www.uttarainfo.com

Uttara Practicals 3

Instructions:

Apply the steps to program discussed for every problem you solve today. Make sure you focus on the process. I have not given the example values & ranges for all the problems - you can consider them as ints.

Steps to program:

a) Capture problem statement

b) Build test cases -> When doing so, come up with number of inputs, ask questions on what type of data, what range of data and then come up with success, boundary, failure input/output cases.

c) Build the method signature (ret type+method name+arg list)

d) Build the pseudocode. This means write down the steps of how you know to solve the problem on paper in english statements. Start with failure case checks first, then single case and post that only apply the business logic.

e) Implement the pseudocode as code! Convert english statements to actual code in programming language. Keep track of how datatypes, operators, control statements work during this translation.

f) Test the code using the test cases.

g) Refactor to improve the quality of the code!

Code all these methods as static methods and invoke from main() to test it. OOAD is not required for these.

Programming Problems:

1) WAP to test if a given number is even (use % first and then and-ing with 1 using bitwise & operator to test)

2) Accept 2 ints, print the multiplication tables of the smaller number till the second number times (if 5,2 and given, print 2X1 till 2X5) => Print directly inside the method

3) WAP to test if a given number is divisible by 3. Do this for both +ve and -ve ints.

4) WAP to test if a given number is prime.

5) WAP to generate all primes within an positive int number given. Can you reuse the method coded for 4th problem?

6) Given 3 numbers, return the correct average of them.

7) WAP to test number of bits set to 1 in a positive int given.

```
int num = <value>;

int count = 0;

for(int i = 0 ; i < 31 ; i++)

{

        if((num & 1) == 1) //do you understand this?

                count++;

        num = num >> 1;

}

return count;
```

8) WAP to test if a given int number is a positive power of 2.

Slightly more complex:

9) WAM to test if all digits in a given number are in decreasing order (allow -ves).

10) WAM to test if all digits in a given number are in increasing order (allow -ves).

11) WAM to test if all digits in a given number are the same (allow -ves).

12) WAM to return the sum of all the digits in a given number (allow -ves).

13) WAM to return the if a given digit is present in a given number.

14) WAP to identify if an int number is a palindrome! Generate all palindromes from 10 till 1 million and print to monitor.

15) WAP to test whether 2 numbers given as inputs contains the same digits, for ex: 121, 112, 211 contain the same digits!

16) WAP to test GCD of 2 int numbers given as parameter.

17) Given 3 numbers, return the second biggest.