Midterms Cheatsheet

Questions with Multiple Options

- "Which of the following.."

Modulo Formula

- x % n = x - ((x / n) * n)

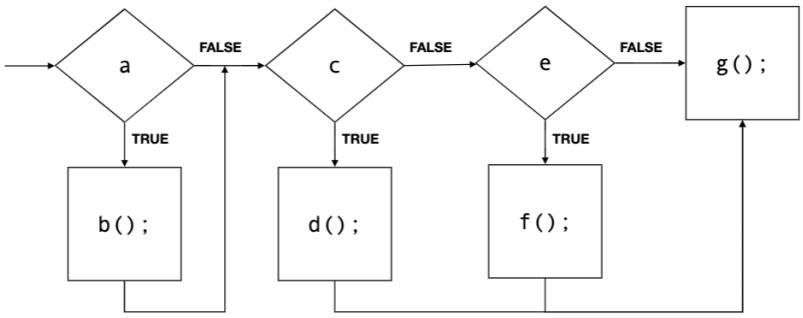
Recursive Functions

-

  Qn: Returns true if m is subnumber of n

  - Last digit matches and m/10 is subnumber of n/10. (eg: 123 subnumber of 9123)
  - Last digit doesn't match but m is subnumber of n/10. (eg: 123 subnumber of 1237)
  - Base case:
    - 0 is subnumber of all integers
    - If m is not 0, then if n == 0, m is not subnumber of n.

Loops



- Corresponding C code:

```
if (a) {
  b();
}
if (c) {
  d();
} else if (e) {
  f();
```

```
}
g();
```

Which loops correctly compute the sum of all integers b/w i and j inclusive?

```
long foo(long i, long j) {
  long sum = i;
    long x = i;
    while (x < j) { //Ends at j-1
      x += 1; //INCREMENTS X BEFORE ADDING
        sum += x;

    }
  return sum;
}

long bar(long i, long j) {
  long sum = 0;
    long x = i;
    do {
      sum += x;
        x += 1;
    } while (x <= j);
  return sum;
}

long qux(long i, long j) {
  long sum = 0;
    for (long x = j; x >= i; x = 1) {
      sum += x;
    }
  return sum;
}
```

All loops will compute the sum of all integers b/w i and j inclusive.

Integer Division

```
double x = (1/2) * (2/3) * (3/4);
```

What's the value of x?

- 0: All the division results in 0.

How to solve it?

- Typecast either or both values to double so proper division occurs.

Imprecision due to Floating pt numbers

```
//change is of double type
while (change > 0) {
    if (change > 1) {
    one_dollar += 1;
    change = 1.0;
    } else if (change > 0.5) {
        fifty_cent += 1;
    change = 0.5;
    } else if (change > 0.2) {
        twenty_cent += 1;
        change = 0.2;
    } else if (change > 0.1) {
        ten_cent += 1;
        change = 0.1;
    }
}
```

- The > operators should instead be operators.
- Imprecisions using floating-point values might mean that certain conditions may be evaluated incorrectly, and more importantly, that one may never reach change == 0

De Morgan's Law

- Qn: Married if and only if either (i) he or she is at least 21 years old, or (ii) he or she is at least 16 years old and have the consent of the parents.

  Write function `cannot_get_married`

Short-Circuiting

- Do cheaper operation first!

Qn: Write a recursive function has8 in C that, given an positive integer number, returns true if the digit 8 appears somewhere in the number, returns false otherwise. Your code should consist of a single return statement and appropriate use short-circuiting to avoid unnecessary checks.

Ans:

- `return (x % 10 == 8) || (x > 10 && has8(x/10));`
  - Since recursive calls use more resources, we should call `x % 10 == 8` condition first to allow short-circuiting.

Loop Invariant

- Assertion that's true at the following points:

  - Before the loop
  - After each iteration of loop
  - After the loop
- Always explain effects on variable in words
  - Eg: i += 1 –> i is incremented by 1 but the rest of the variables remain the same so we decrement i by 1, thus

$$ya^{i-1} == x^n$$

    .
  - Eg:

$$i = (i-1)/2$$

    . i is decremented and halved, but the rest of the variables remain the same. So we multiply i by 2 and increment it by 1, thus

$$ya^{2i+1} == x^n$$

- Since i % 3 == 0 , (i/3) is the same as (i-1)/3 + 1

Is_prime Algorithm

```
//Correct Algo (given n >= 2)
bool is_prime(long n) {
    for (long i = 2; i <= sqrt(n); i += 1) {
        if ((n % i) == 0) {
            return false;
        }
    }
    return true;
}

//Wrong as algo does not check for the case where i is sqrt(n).
//Counter-example: Number that is a square of a prime (eg: 4, 9, 25, ...) will be incorrect
bool is_prime(long n) {
    for (long i = 2; i < sqrt(n); i += 1) {
        if ((n % i) == 0) {
            return false;
        }
    }
    return true;
}

//Wrong as function would return 2 as a non-prime.
bool is_prime(long n) {
    for (long i = 2; i <= ceil(sqrt(n)); i += 1) {
        if ((n % i) == 0) {
            return false;
```

```
            }
        }
        return true;
    }
```