Practice (Functions)

Quick questions on functions

1) Sum of first n natural numbers
2) Check pythagorian triplet
3) Conversions
   a) Binary to decimal
   b) Octal to decimal
   c) Hexadecimal to decimal
   d) Decimal to binary
   e) Decimal to octal
   f) Decimal to hexadecimal
4) Add two binary numbers

$x2^3+\dots+y2^2+z2^1+w2^0$

$x8^n+\dots+y8^1+z8^0$

$x16^n+\dots+y16^1+z16^0$

$(\ )_{10}$ $\div 2$ → Remainders in reverse order.

$\div 8$ → "

$\div 16$ → "

Sum of first n natural numbers

$$S = 1 + 2 + 3 + 4 \dots n$$

$$= \frac{n(n+1)}{2}$$

Check pythagorian triplet

Given : x, y, z

Let $a = \max(x, y, z)$, and b and c be the other numbers

Idea : $a^2 = b^2 + c^2$

Let's code them...

Apni Kaksha

Conversions

Binary to decimal

— Covered in video 5.1

Octal to decimal

Base: 8

{0, 1, 2, 3, 4, 5, 6, 7}.

Representation: $(x)_8$

Eg. $(137)_8$

Converting to Decimal

$$(137)_8 = 7 \times 8^0 + 3 \times 8^1 + 1 \times 8^2$$

$$= 7 + 24 + 64$$

$$= (95)_{10}$$

Algorithm: Traverse over the digits and make the decimal number.

Hexadecimal to decimal

Base : 16

{0, 1, 2, 3, 4, 5, 6 .... 9, A, B, C, D, E, F}

A = 10	D = 13
B = 11	E = 14
C = 12	F = 15

Eg $(1CF)_{16}$

Converting to Decimal

$(1CF)_{16} = 15 \times 16^0 + 12 \times 16^1 + 1 \times 16^2$

$= 15 + 192 + 256$

$= (463)_{10}$

Algorithm: Traverse over the digits and make the decimal number.

Decimal to binary → ?

- Covered in video 5.1

Decimal to octal

Eg. $(100)_{10} = (x)_8$

$(100)_{10} = 1 \times 8^2 + 4 \times 8^1 + 4 \times 8^0$

$= (144)_8$

not the sum → sum in Oct to Dec.

Algorithm: Find the highest power of 8, from which our number is divisible, then reduce the number. Do this until our number becomes 0.

Apni Kaksha

Decimal to hexadecimal

Eg $(100)_{10} = (x)_{16}$

$$(100)_{10} = 6 \times 16^1 + 4 \times 16^0$$

$$= (64)_{16}$$

Algorithm: Find the highest power of 16, from which our number is divisible, then reduce the number. Do this until our number becomes 0.

Let's code them...

Add two binary numbers

Given 2 binary numbers

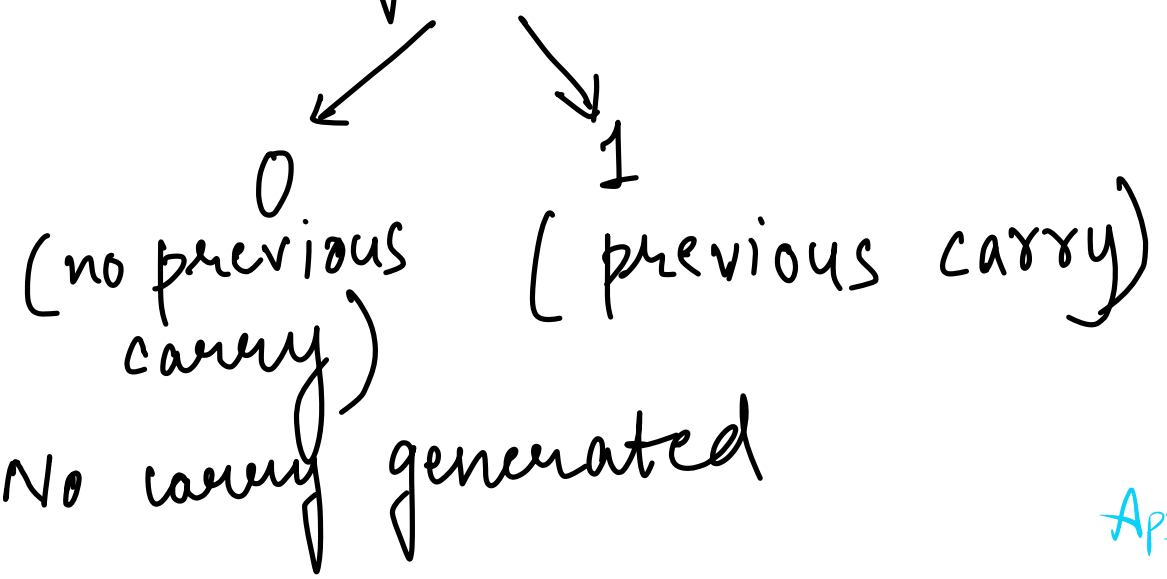
a = 10101

b = 11011

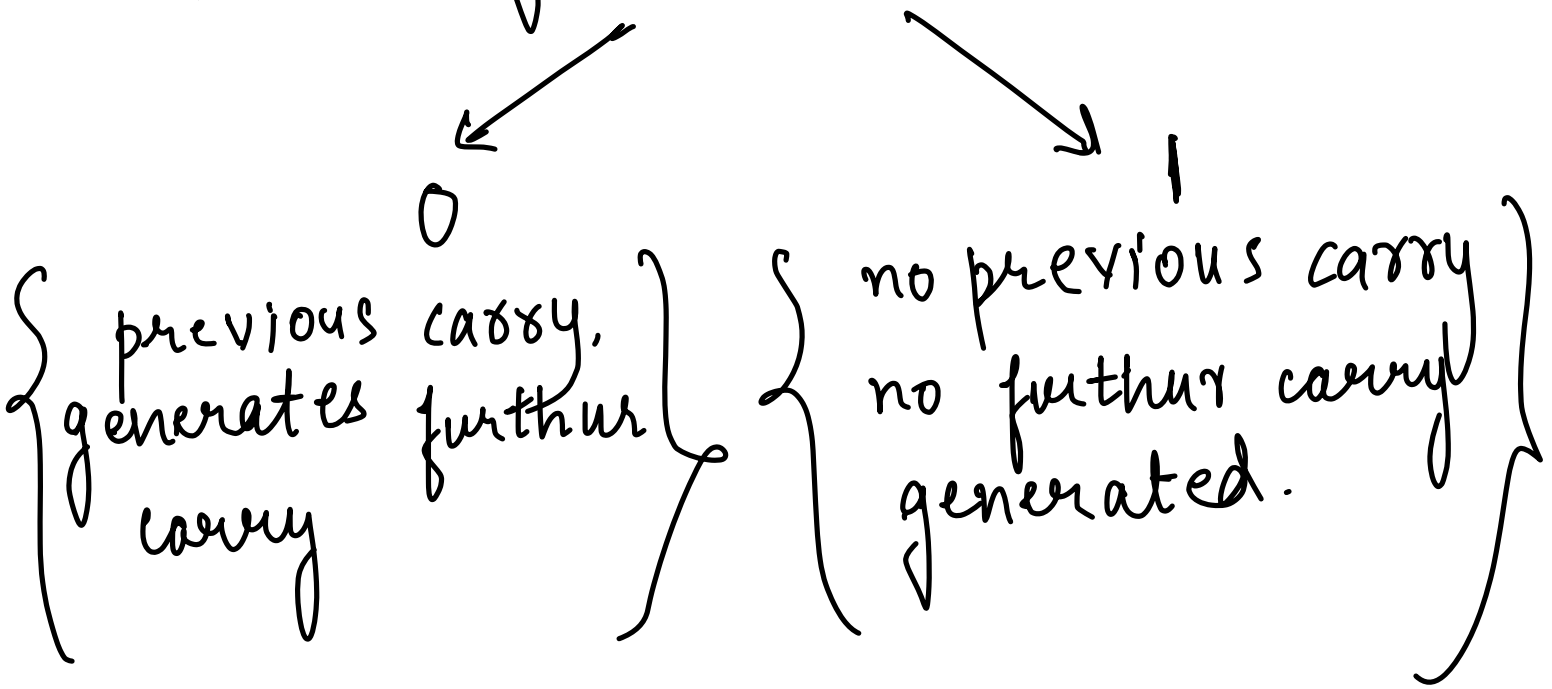
```
  1 1 1 1 1
    1 0 1 0 1
+   1 1 0 1 1
-----------------
  1 1 0 0 0 0
-----------------
```

Algorithm: Traverse digits of both the numbers, there will be three case

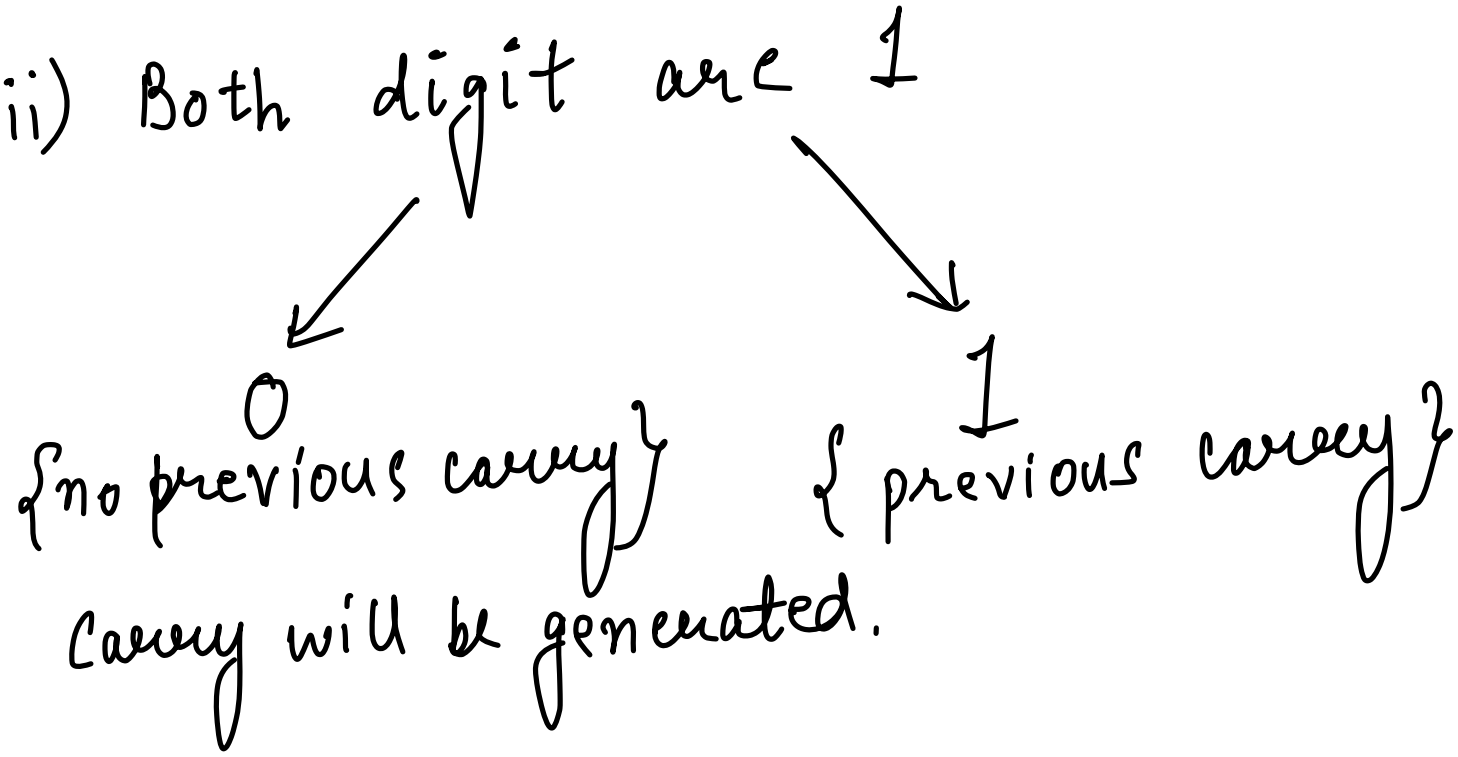
(i) Both digits are 0



(ii) One digit is 0, other is 1



(iii) Both digit are 1



Codes

1. Sum of first n natural numbers

```
int sum(int n)
{
    int ans = 0;
    for(int i=1; i<=n; i++)
        ans += i;

    return ans;
}
```

```
int sum(int n)
{
    return (n*(n+1))/2;
}
```

---

2. Check pythagorean triplet

```
bool check(int x, int y, int z)
{
    int a = max(x, max(y,z));
    int b,c;

    if(a == x) {
        b = y;
        c = z;
    }
    else if(a == y) {
        b = x;
        c = z;
    }
    else {
        b = x;
        c = y;
    }
    if(a*a == b*b + c*c)
        return true;
    return false;
}
```

Conversions

a) Binary to Decimal

```
int binaryToDecimal(int n)
{
    int ans = 0;
    int x = 1;
    while(n>0)
    {
        int y = n%10;
        ans += x*y;
        x *= 2;
        n /= 10;
    }

    return ans;
}
```

b) Octal to decimal

```
int octalToDecimal(int n)
{
    int ans = 0;
    int x = 1;
    while(n>0)
    {
        int y = n%10;
        ans += x*y;
        x *= 8;
        n /= 10;
    }

    return ans;
}
```

c) Hexadecimal to decimals

```
int hexadecimalToDecimal(string n)
{
    int ans = 0;
    int x = 1;

    int s = n.size();
    for(int i=s-1; i>=0; i--)
    {
        if(n[i] >= '0' && n[i] <= '9')
        {
            ans += x*(n[i]-'0');
        }
        else if(n[i] >= 'A' && n[i] <= 'F')
        {
            ans += x*(n[i]-'A' + 10);
        }

        x *= 16;
    }

    return ans;
}
```

d) Decimal to binary

```
int decimalToBinary(int n)
{
    int x = 1;
    int ans = 0;
    while(x <= n)
    {
        x *= 2;
    }
    x/=2;

    while(x>0)
    {
        int lastDigit = n/x;
        n -= lastDigit*x;
        x/=2;
        ans = ans*10 + lastDigit;
    }
    return ans;
}
```

e) Decimal to octal

```
int decimalToOctal(int n)
{
    int x = 1;
    int ans = 0;
    while(x <= n)
    {
        x *= 8;
    }
    x /= 8;

    while(x>0)
    {
        int lastDigit = n/x;
        n -= lastDigit*x;
        x/=8;
        ans = ans*10 + lastDigit;
    }
    return ans;
}
```

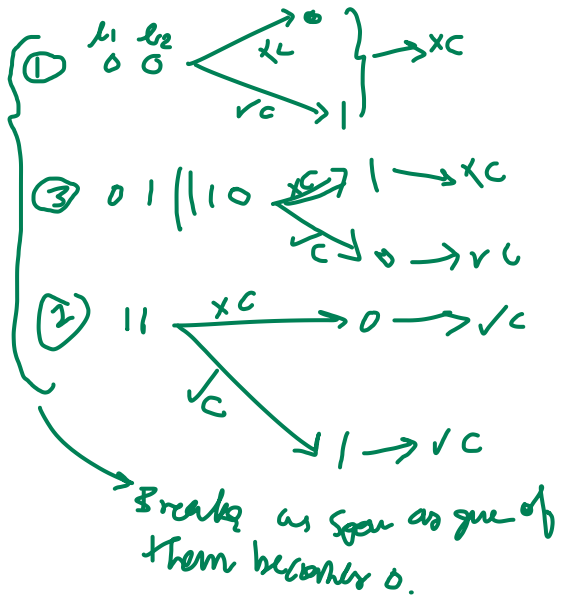
f) Decimal to hexadecimal

```
string decimalToHexadecimal(int n)
{
    int x = 1;
    string ans = "";
    while(x <= n)
    {
        x *= 16;
    }
    x /= 16;

    while(x>0)
    {
        int lastDigit = n/x;
        n -= lastDigit*x;
        x/=16;

        if(lastDigit <= 9)
            ans = ans + to_string(lastDigit);

        else
        {
            char c = 'A' + lastDigit - 10;
            ans.push_back(c);
        }
    }
    return ans;
}
```

4. Add two binary numbers

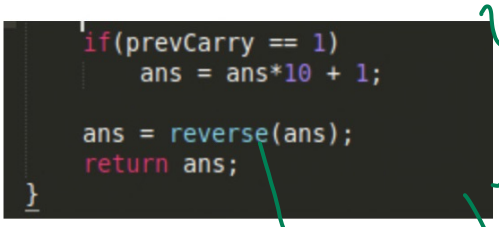
```
int reverse(int n)
{
    int ans = 0;
    while(n>0)
    {
        int lastDigit = n%10;
        ans = ans*10 + lastDigit;
        n /= 10;
    }
    return ans;
}
```

```
int addBinary(int a, int b)
{
    int ans=0; int prevCarry=0;
    while(a>0 && b>0){
        if(a%2 == 0 && b%2 == 0){
            ans = ans*10 + prevCarry;
            prevCarry = 0;
        }
        else if((a%2 == 0 && b%2 == 1) || (a%2 == 1 && b%2 == 0)){
            if(prevCarry == 1){
                ans = ans*10 + 0;
                prevCarry = 1;
            }
            else{
                ans = ans*10 + 1;
                prevCarry = 0;
            }
        }
        else{
            ans = ans*10 + prevCarry;
            prevCarry = 1;
        }
        a /= 10; b /= 10;
    }
```




```
while(a>0){
    if(prevCarry == 1){
        if(a%2 == 1){
            ans = ans*10 + 0;
            prevCarry = 1;
        }
        else{
            ans = ans*10 + 1;
            prevCarry = 0;
        }
    }
    else
        ans = ans*10 + (a%2);
    a /= 10;
}

while(b>0){
    if(prevCarry == 1){
        if(b%2 == 1){
            ans = ans*10 + 0;
            prevCarry = 1;
        }
        else{
            ans = ans*10 + 1;
            prevCarry = 0;
        }
    }
    else
        ans = ans*10 + (b%2);
    b /= 10;
}
```

For the bigger number.



when both of them were equal, works but at end the carry was generated but, ∵ both a & b became 0, so carry remain used.

to use that one carry.

Func" by use

↳ ∵ digits were appended → it got in reverse so to correct it.