Math Handbook

Cody Barnson

January 15, 2019

Contents

1 Math

1.0.1 Limits

$$\lim_{a \to \infty} \frac{1}{a} \tag{1}$$

$$\lim_{a \to \infty} \frac{1}{a} \tag{2}$$

1.0.2 Log base conversion

$$\frac{log_x n}{log_x B} = log_B n$$

1.0.3 Ceiling integer division

$$\left\lfloor \frac{n}{d} \right\rfloor = \frac{n + d - 1}{d}$$

1.0.4 Bit shift equivalent of multiply by 10

```
// (x << 3) + (x << 1) ≡ x * 10
int x, y;
// ...
x = (x << 3) + (x << 1);
y = y * 10;
assert(x == y);
```

2 Programming

2.1 C++

2.1.1 Logarithm base 2

```
// log2(n)
log2(n) = 31 - __builtin_clz(n);
```

2.1.2 Add value, update average

```
// avg_{n+1} = (sum_n(n+1) + kn - sum) / (n(n+1))
int n, sum;
// ...
double avg = sum / n;
while ((int)(avg + 0.5) < k) {
    avg = sum * n + sum + k * n - sum;
    avg /= n * n + n;
    sum += k;
    n++;
}
```

2.1.3 Binomial coefficient

```
// C(n,k) = C(n-1,k-1) + C(n-1,k)
typedef long long ll;
ll binom(int n, int k) {
    if (k == 0 || k == n) return 1;
    k = min(k, n - k); // Since C(n,k) ≡ C(n,n-k)
    ll ans = 1LL;
    for (ll i = 1; i <= k; i++) {
        ans = ans * (n - k + i) / i;
    }
}
ll choose(int n, int k, ll p = 1e9+7) {
    if (n < k) return 0;
    k = min(k, n - k);
    ll num = 1, den = 1;
    for (int i = 0; i < k; i++) num = num * (n - i) % p;
    for (int i = 1; i <= k; i++) den = den * i % p;
    return num * powmod(den, p - 2, p) % p;
}
ll multichoose(int n, int k, ll p = 1e9+7) {
    return choose(n + k - 1, k, p);
}
```

2.1.4 Catalan numbers

```
typedef long long ll;
ll catalan(int n, ll p = 1e9+7) {
    return choose(2 * n, n, p) * powmod(n + 1, p - 2, p) % p;
}
ll powmod(ll x, ll n, ll m) {
    ll a = 1, b = x;
    for (; n > 0; n >>= 1) {
        if (n & 1) a = mulmod(a, b, m);
        b = mulmod(b, b, m);
    }
    return a % m;
}
ll mulmod(ll x, ll n, ll m) {
    ll n = 0, b = x % m;
    for (; n > 0; n >>= 1) {
        if (n & 1) a = (a + b) % m;
        b = (b << 1) % m;
    }
    return a % m;
}
```

2.1.5 Count number of digits in a number

```
// digits = ⌊log10(n)⌋ + 1
int countDigits(long long n) {
    return n > 0 ? (int)log10((double)n) + 1 : 1;
}
```

2.1.6 Enumerate combinations of N elements in K in lexical order

```
typedef long long ll;
ll catalan(int n, ll p = 1e9+7) {
    return choose(2 * n, n, p) * powmod(n + 1, p - 2, p) % p;
}
ll powmod(ll x, ll n, ll m) {
    ll a = 1, b = x;
    for (; n > 0; n >>= 1) {
        if (n & 1) a = mulmod      b = mulmod(b, b, m);
    }
    return a % m;
}
ll mulmod(ll x, ll n, ll m) {
    ll n = 0, b = x % m;
    for (; n > 0; n >>= 1) {
        if (n & 1) a = (a + b) % m;
        b = (b << 1) % m;
    }
    return a % m;
}
```

2.1.7 Prime factorization

```
typedef vector<int> vi;
vi factor(int n) {
    vi f;
    if (n < 2) return vi();
    while (~n & 1) n /= 2, f.push_back(2);
    for (long long p = 3; p * p <= n; p += 2)
        while (n % p == 0) n /= p, f.push_back((int)p);
    if (n > 1) f.push_back(n);
    return f;
}
```

2.1.8 Fibonacci

```
// Complexity: O(log(n))
// Compute x^n mod m
int modexp(int x, int n, int m) {
  if (n == 0) return 1;
  if (n & 1) return ((x % m) * modexp(x, n - 1, m)) % m;
  int y = modexp(x, n / 2, m);
  return (y * y) % m;
}
```

2.1.9 Modular Exponentiation

```
// Complexity: O(log(n))
// Compute x^n mod m
int modexp(int x, int n, int m) {
  if (n == 0) return 1;
  if (n & 1) return ((x % m) * modexp(x, n - 1, m)) % m;
  int y = modexp(x, n / 2, m);
  return (y * y) % m;
}
```

2.1.10 Sieve + Optimized primality testing

```
// Sieve + optimized prime testing
typedef long long ll;
typedef vector<int> vi;

ll sz;
bitset<10000010> p; // 10^7 + 10
vi primes;
void sieve(ll m) {
    sz = m + 1;
    p.set();
    p[0] = p[1] = 0;
    for (ll i = 2; i <= sz; i++) {
        if (p[i]) {
            for (ll j = i * i; j <= sz; j += i) {
                p[j] = 0;
            }
            primes.push_back((int)i);
        }
    }
}
bool isPrime(ll x) {
    if (x <= sz) return p[x];
    for (int i = 0; i < (int)primes.size(); i++) {
        if (x % primes[i] == 0) return false;
    }
    return true;
}
```

2.1.11 Base conversion

```cpp
// Base conversion
// Complexity: O(N), N digits
// Given digits of int x in base a, return x's digits in base b.

typedef vector<int> vi;

// x : digit representation of number
// a : base of x
// b : desired base
// returns => vector<int> digits of number in base b.
// Note: vec[0] stores the most significant digit.
vi convert_base(const vi &x, int a, int b) {
    unsigned long long base10 = 0;
    FR(i, x.size()) base10 += x[i] * pow(a, x.size() - i - 1);
    int N = ceil(log(base10 + 1) / log(b));
    vi bb;
    for (int i = 1; i <= N; i++)
        bb.emplace_back((int)(base10 / pow(b, N - i)) % b);
    return bb;
}

// x : number
// b : desired base
// returns => vector<int> digits of number in base b
vi base_digits(int x, int b = 10) {
    vi bb;
    while (x != 0) bb.emplace_back(x % b), x /= b;
    reverse(begin(bb), end(bb));
    return bb;
}

int main() {
    // consider 123_5, (i.e. 123 in base 5)
    vi x{1, 2, 3}; int a = 5;
    vi z = convert_base(x, a, 10); // 123_5 = 38_10, z = {3, 8}
    vi y = convert_base(x, a, 3); // 123_5 = 1102_3, y = {1, 1, 0, 2}
}
```

3 LaTeX

3.1 Learning Resources

- LaTeX Wiki
- Detexify
- The Comprehensive LaTeX Symbol List
- LaTeX Utility
- LaTeX Primer
- TeX Primitive Control Sequences

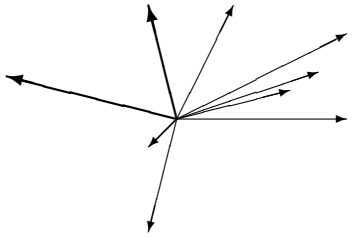
- Cambridge Text Processing using LaTeX

3.2 Picture

For more information, see Picture. Also, see Gnuplot.

Draw arrows

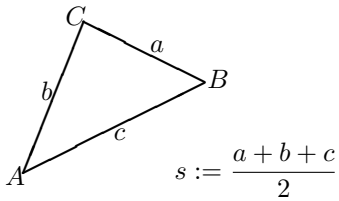
```
\setlength{\unitlength}{0.75mm}
\begin{picture}(60,40)
    \put(30,20){\vector(1,0){30}}
    \put(30,20){\vector(4,1){20}}
    \put(30,20){\vector(3,1){25}}
    \put(30,20){\vector(2,1){30}}
    \put(30,20){\vector(1,2){10}}
    \thicklines
    \put(30,20){\vector(-4,1){30}}
    \put(30,20){\vector(-1,4){5}}
    \thinlines
    \put(30,20){\vector(-1,-1){5}}
    \put(30,20){\vector(-1,-4){5}}
\end{picture}
```



Text and formulae with `put` command

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,5)
    \thicklines
    \put(1,0.5){\line(2,1){3}}
    \put(4,2){\line(-2,1){2}}
    \put(2,3){\line(-2,-5){1}}
    \put(0.7,0.3){$A$}
    \put(4.05,1.9){$B$}
    \put(1.7,2.95){$C$}
    \put(3.1,2.5){$a$}
    \put(1.3,1.7){$b$}
    \put(2.5,1.05){$c$}
    \put(0.3,4){$F=\sqrt{s(s-a)(s-b)(s-c)}$}
    \put(3.5,0.4){$\displaystyle s:=\frac{a+b+c}{2}$}
\end{picture}
```

$$F = \sqrt{s(s-a)(s-b)(s-c)}$$



$$s := \frac{a+b+c}{2}$$

3.3 Mathematics

TODO: Mathematics

3.4 Document Structure

TODO: Document Structure

3.5 Footnotes and Margin Notes

TODO: Footnotes and Margin Notes

3.5.1 Creating a footnote

This is a footnote.[1]

3.6 Package Reference

TODO: Package Reference

[1] And here is the footnote text.

3.7 Font

3.7.1 Font sizes

texblog.org	\Huge
texblog.org	\huge
texblog.org	\LARGE
texblog.org	\Large
texblog.org	\large
texblog.org	\normalsize
texblog.org	\small
texblog.org	\footnotesize
texblog.org	\scriptsize
texblog.org	\tiny

Table 1: Font sizes

3.8 TikZ

3.8.1 Drawing lines

TODO: TikZ

All drawing commands inside a tikzpicture environment:

All drawing commands inside tikzpicture

```
\begin{tikzpicture}
    \draw (1,0) -- (0,0) -- (0,1);
\end{tikzpicture}
```

Connected path closed with `--cycle` operation

```
\draw (1,0) -- (0,0) -- (0,1) -- cycle;
```

Move-to operation

```
\draw (0,0) -- (2,0) (0,1) -- (2,1);
```

Connect points by first horizontal then vertical

```
\draw (0,0) -| (1,1);
```

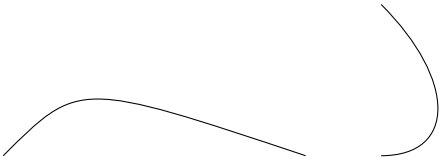
Connect points by first vertical then horizontal

```
\draw (0,0) |- (1,1);
```

3.8.2 Drawing curved paths

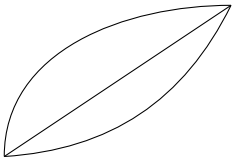
Bezier curve using `controls` command

```
\draw (0,0) .. controls (1,1) .. (4,0)
(5,0) .. controls (6,0) and (6,1) .. (5,2);
```



User-defined paths using the `to` operation

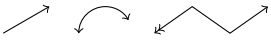
```
\draw (0,0) to (3,2);
\draw (0,0) to[out=90,in=180] (3,2);
\draw (0,0) to[bend right] (3,2);
```



3.8.3 Special curves

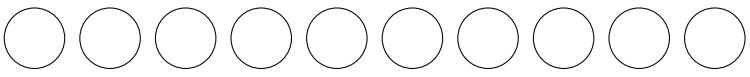
Arrow tips to lines

```
\draw [->] (0,0) -- (30:20pt);
\draw [<->] (1,0) arc (180:30:10pt);
\draw [<<->] (2,0) -- ++(0.5,10pt) -- ++(0.5,-10pt) -- ++(0.5,10pt);
```



Loop using `foreach` command

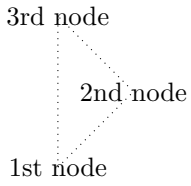
```
\foreach \x in {0,...,9}
\draw (\x,0) circle (0.4);
```



3.8.4 Nodes

Draw text along path with node command

```
\draw[dotted]
(0,0) node {1st node}
-- (1,1) node {2nd node}
-- (0,2) node {3rd node}
-- cycle;
```



Connect nodes using nodes' labels as coordinates (Option 1)

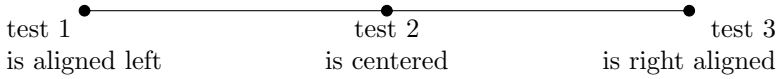
```
\path (0,0) node(x) {}
(3,1) node(y) {};
\draw (x) -- (y);
```

Connect nodes using nodes' labels as coordinates (Option 2)

```
\coordinate (x) at (0,0);
\coordinate (y) at (3,1);
\draw (x) -- (y);
```

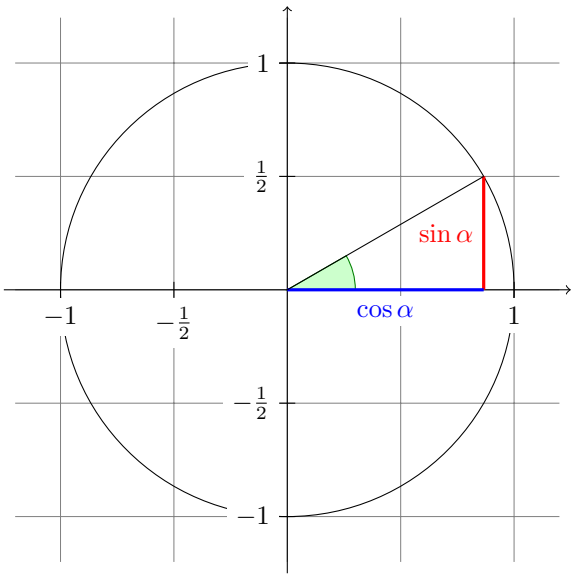
Multi-line text inside a node

```
\filldraw
(0,0) circle (2pt) node[align=left,   below] {test 1\\is aligned left} --
(4,0) circle (2pt) node[align=center, below] {test 2\\is centered}     --
(8,0) circle (2pt) node[align=right,  below] {test 3\\is right aligned};
```



For inline

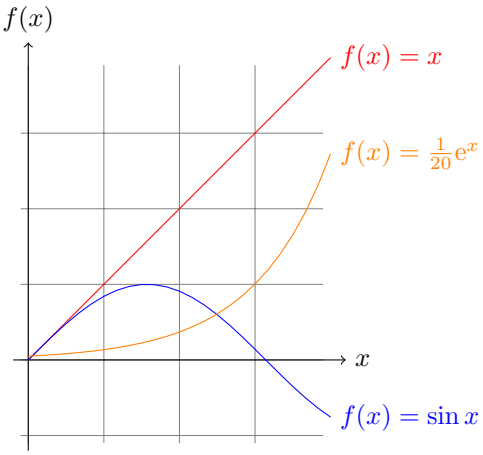
```
\tikz[]{\draw[red, dashed, very thick, rotate=30] (1,0) -- (0,0) -- (0,1);}
```



Unit circle on graph

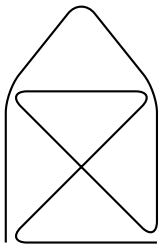
Functions

```
\begin{tikzpicture}[domain=0:4]
    \draw[very thin,color=gray] (-0.1,-1.1) grid (3.9,3.9);
    \draw[->] (-0.2,0) -- (4.2,0) node[right] {$x$};
    \draw[->] (0,-1.2) -- (0,4.2) node[above] {$f(x)$};
    \draw[color=red]    plot (\x,\x)             node[right] {$f(x) =x$};
    \draw[color=blue]   plot (\x,{sin(\x r)})    node[right] {$f(x) = \sin x$};
    \draw[color=orange] plot (\x,{0.05*exp(\x)}) node[right] {$f(x) = \frac{1}{20} \mathrm e^x$};
\end{tikzpicture}
```



Shape with rounded corners

```
\begin{tikzpicture}
    \draw[thick,rounded corners=8pt] (0,0) -- (0,2) -- (1,3.25)
    -- (2,2) -- (2,0) -- (0,2) -- (2,2) -- (0,0) -- (2,0);
\end{tikzpicture}
```



3.9 Letters

TODO: Letters

3.10 Source code

TODO: Source code listings

3.11 Comments on equations

3.11.1 Comment on parts

$$z = \overbrace{\underbrace{x}_{\text{real}} + i \underbrace{y}_{\text{imaginary}}}^{\text{complex number}}$$

3.11.2 Comments longer than formula

$$y = a + f(\underbrace{bx}_{\geq 0 \text{ by assumption}}) = a + f(\underbrace{bx}_{\geq 0 \text{ by assumption}})$$