The whole world runs on numbers. So protect your information yourself.

- **Linux Command and C –**
- **36 pages**

- Linux Command List : -

```
 ----------------------
| Hardware Information  |
 ----------------------
```

- show bootup message:

```
dmesg
```

- See CPU information:

```
cat /proc/cpuinfo
```

- Display free and used memory with:

```
free -h
```

- List hardware configuration information:

```
lshw
```

- See information about block devices:

```
lsblk
```

- Show PCI devices in a tree-like diagram:

```
lspci -tv
```

- Display USB devices in a tree-like diagram:

```
lsusb -tv
```

- Show hardware information from the BIOS:

```
  dmidecode
```

- Display disk data information:

```
  hdparm -i /dev/disk
```

- Conduct a read-speed test on device/disk:

```
  hdparm -tT /dev/[device]
```

- Test for unreadable blocks on device/disk:

```
  badblocks -s /dev/[device]
```

| Searching:|

- Search for a specific pattern in a file with grep:

```
  grep [pattern] [file_name]
```

- Recursively search for a pattern in a directory:

```
  grep -r [pattern] [directory_name]
```

- Find all files and directories related to a particular name:

```
  locate [name]
```

- List names that begin with a specified character [a] in a specified location [/folder/location] by using the find command:

```
  find [/folder/location] -name [a]
```

- See files larger than a specified size [+100M] in a folder:

```
  find [/folder/location] -size [+100M]
```

|File Commands:-|

- List files in the directory:

```
  ls
```

- List all files (shows hidden files):

```
  ls -a
```

- Show directory you are currently working in:

```
  pwd
```

- Create a new directory:

```
  mkdir [directory]
```

- Remove a file:

```
  rm [file_name]
```

- Remove a directory recursively:

```
  rm -r [directory_name]
```

- Recursively remove a directory without requiring confirmation:

```
  rm -rf [directory_name]
```

- Copy the contents of one file to another file:

```
  cp [file_name1] [file_name2]
```

- Recursively copy the contents of one file to a second file:

```
cp -r [directory_name1] [directory_name2]
```

- Rename [file_name1] to [file_name2] with the command:

```
mv [file_name1] [file_name2]
```

- Create a symbolic link to a file:

```
ln -s /path/to/[file_name] [link_name]
```

- Create a new file:

```
touch [file_name]
```

- Show the contents of a file:

```
more [file_name]
```

- or use the cat command:

```
cat [file_name]
```

- Append file contents to another file:

```
cat [file_name1] >> [file_name2]
```

- Display the first 10 lines of a file with:

```
head [file_name]
```

- Show the last 10 lines of a file:

```
tail [file_name]
```

- Encrypt a file:

gpg -c [file_name]

- Decrypt a file:

gpg [file_name.gpg]

- Show the number of words, lines, and bytes in a file:

wc

|Directory Navigation:-|

- Move up one level in the directory tree structure:

  cd ..

- Change directory to $HOME:

  cd

- Change location to a specified directory:

cd /chosen/directory

|File Compression:-|

- Archive an existing file:

  tar cf [compressed_file.tar] [file_name]

- Extract an archived file:

```
  tar xf [compressed_file.tar]
```

- Create a gzip compressed tar file by running:

```
  tar czf [compressed_file.tar.gz]
```

- Compress a file with the .gz extension:

```
  gzip [file_name]
```

|File Transfer:-|

- Copy a file to a server directory securely:

```
  scp [file_name.txt] [server/tmp]
```

- Synchronize the contents of a directory with a backup directory using the rsync command:

```
  rsync -a [/your/directory] [/backup/]
```

|Users:-|

- See details about the active users:

```
  id
```

- Show last system logins:

```
  last
```

- Display who is currently logged into the system with the who command:

```
 who
```

- Show which users are logged in and their activity:

```
 w
```

- Add a new group by typing:

```
 groupadd [group_name]
```

- Add a new user:

```
 adduser [user_name]
```

- Add a user to a group:

```
 usermod -aG [group_name] [user_name]
```

- Temporarily elevate user privileges to superuser or root using the sudo command:

```
 sudo [command_to_be_executed_as_superuser]
```

- Delete a user:

userdel [user_name] | *Eight Multiply Four =< Page Num >*

- Modify user information with:

```
 usermod
```

|Package Installation:-|

- List all installed packages with yum:

```
 yum list installed
```

- Find a package by a related keyword:

```
yum search [keyword]
```

- Show package information and summary:

```
yum info [package_name]
```

- Install a package using the YUM package manager:

```
yum install [package_name.rpm]
```

- Install a package using the DNF package manager:

```
dnf install [package_name.rpm]
```

- Install a package using the APT package manager:

```
apt-get install [package_name]
```

- Install an .rpm package from a local file:

```
rpm -i  [package_name.rpm]
```

- Remove an .rpm package:

```
rpm -e [package_name.rpm]
```

- Install software from source code:

```
tar zxvf [source_code.tar.gz]
cd [source_code]
./configure
make
make install
```

```
------------------
|Process Related:-|
------------------
```

- See a snapshot of active processes:

```
 ps
```

- Show processes in a tree-like diagram:

```
pstree
```

- Display a memory usage map of processes:

```
pmap
```

- See all running processes:

```
top
```

- Terminate a Linux process under a given ID:

```
kill [process_id]
```

- Terminate a process under a specific name:

```
pkill [proc_name]
```

- Terminate all processes labelled “proc”:

```
killall [proc_name]
```

- List and resume stopped jobs in the background:

```
bg
```

- Bring the most recently suspended job to the foreground:

fg

- Bring a particular job to the foreground:

fg [job]

- List files opened by running processes:

lsof

|System Information:-|

- Show system information:

uname -r

- See kernel release information:

uname -a

- Display how long the system has been running, including load average:

uptime

- See system hostname:

hostname

- Show the IP address of the system:

hostname -i

- List system reboot history:

last reboot

- See current time and date:

```
date
```

- Query and change the system clock with:

```
timedatectl
```

- Show current calendar (month and day):

```
cal
```

- List logged in users:

```
w
```

- See which user you are using:

```
whoami
```

- Show information about a particular user:

```
finger [username]
```

|Disk Usage:-|

- You can use the df and du commands to check disk space in Linux.

- See free and used space on mounted systems:

```
df -h
```

- Show free inodes on mounted filesystems:

```
df -i
```

- Display disk partitions, sizes, and types with the command:

```
fdisk -l
```

- See disk usage for all files and directory:

```
du -ah
```

- Show disk usage of the directory you are currently in:

```
du -sh
```

- Display target mount point for all filesystem:

```
findmnt
```

- Mount a device:

```
mount [device_path] [mount_point]
```

|SSH Login:-|

- Connect to host as user:

```
ssh user@host
```

- Securely connect to host via SSH default port 22:

```
ssh host
```

- Connect to host using a particular port:

```
ssh -p [port] user@host
```

- Connect to host via telnet default port 23:

```
telnet host
```

|File Permission:-|

- Chown command in Linux changes file and directory ownership.

- Assign read, write, and execute permission to everyone:

```
chmod 777 [file_name]
```

- Give read, write, and execute permission to owner, and read and execute permission to group and others:

```
chmod 755 [file_name]
```

- Assign full permission to owner, and read and write permission to group and others:

```
chmod 766 [file_name]
```

- Change the ownership of a file:

```
chown [user] [file_name]
```

- Change the owner and group ownership of a file:

```
chown [user]:[group] [file_name]
```

|Network:-|

- List IP addresses and network interfaces:

```
ip addr show
```

- Assign an IP address to interface eth0:

```
ip address add [IP_address]
```

- Display IP addresses of all network interfaces with:

```
ifconfig
```

- See active (listening) ports with the netstat command:

```
netstat -pnltu
```

- Show tcp and udp ports and their programs:

```
netstat -nutlp
```

- Display more information about a domain:

```
whois [domain]
```

- Show DNS information about a domain using the dig command:

```
dig [domain]
```

- Do a reverse lookup on domain:

```
dig -x host
```

- Do reverse lookup of an IP address:

```
dig -x [ip_address]
```

- Perform an IP lookup for a domain:

host [domain]

- Show the local IP address:

hostname -I

- Download a file from a domain using the wget command:

wget [file_name]

|Linux Keyboard Shortcuts:-|

- Kill process running in the terminal:

Ctrl + C

- Stop current process:

Ctrl + Z

- The process can be resumed in the foreground with fg or in the background with bg.
- Cut one word before the cursor and add it to clipboard:

Ctrl + W

- Cut part of the line before the cursor and add it to clipboard:

Ctrl + U

- Cut part of the line after the cursor and add it to clipboard:

Ctrl + K

- Paste from clipboard:

Ctrl + Y

- Recall last command that matches the provided characters:

Ctrl + R

- Run the previously recalled command:

Ctrl + O

- Exit command history without running a command:

Ctrl + G

- Run the last command again:

!!

- Log out of current session:

Exit

```
#include <stdio.h>

int check_stair_case(int n) {
        if(n<0) return -1;
        if(n==0) return 0;
        int rows = 0;
        long sum = 0;
        while(sum<=n){
            sum += rows+1;
            rows++;
        }
        return rows-1;
    }

int main(void)
{
  int n = 5;
  printf("Input number %d ",n);
  printf("\nTotal number of full staircase rows are %d
",check_stair_case(n));
  n = 8;
  printf("\nInput number %d ",n);
  printf("\nTotal number of full staircase rows are %d
",check_stair_case(n));
  return 0;
}
```

```
#include <stdio.h>

int find_Nth_Digit(int n) {
    unsigned int i, j, k;
    i = j = 1;
    while (n > 9 * i * j) {
        n -= 9 * i * j;
        j *= 10;
        i ++;
    }
    k = j + (n - 1) / i;
    for (j = (n - 1) % i; j < i - 1; j ++) {
        k = k / 10;
    }
    return k % 10;
}

int main(void)
{
  int n = 7;
  printf("\n%d digit of the sequence is %d",n,find_Nth_Digit(n));
  n = 12;
  printf("\n%d digit of the sequence is %d",n,find_Nth_Digit(n));
  return 0;
}
```

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

void print_lexicographic(int n)
{
        int m, j, i = 1;
        printf("\n\nPrint numbers from 1 to %d in lexicographic order-\n",n);
  while(i<= 9){
    j = 1;
    while( j <= n){
      m = 0;
                    while(m < j) {
        if((m + j * i)<= n){
          printf("%d ", m + j * i);
        }
        m=m+1;
                     }
      j= j*10;
    }
    i=i+1;
  }
}

int main(void)
{
  print_lexicographic(10);
  print_lexicographic(25);
  print_lexicographic(40);
  print_lexicographic(100);
  return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>

int* count_Bits(int num, int* returnSize) {
    int *p, i;

    p = malloc((num + 1) * sizeof(int));
   *returnSize = num + 1;

    p[0] = 0;
    for (i = 1; i <= num; i ++) {
        p[i] = p[i & (i - 1)] + 1;
    }

    return p;
}
```

```
int main(void)
{
   int *p;
   int returnSize;
   int i = 7;
   printf("Number: %d",i);
   printf("\nNumber of 1's in the binary representation:\n");
           p = count_Bits(5, &returnSize);
   for (i = 0; i < returnSize; i++) {
      printf("%X:\t%d\n", i,  p[i]);
   }
   free(p);
      return 0;
}
```

```
#include <stdio.h>
#include <stdbool.h>

static bool is_PowerOf_Three(int n) {
#if 0
   if (n == 1) return true;
   if (n == 0 || n % 3) return false;
   return is_PowerOf_Three(n / 3);
#else
```

```
    return (n > 0 && (1162261467 % n) == 0);
#endif
}
int main(void)
{
    int n = 9;
    printf("\nIf %d is power of three? %d", n, is_PowerOf_Three(n));
    n = 81;
    printf("\n\nIf %d is power of three? %d", n, is_PowerOf_Three(n));
    n = 45;
    printf("\n\nIf %d is power of three? %d", n, is_PowerOf_Three(n));
    return 0;
}
```

```
#include <stdio.h>

static int addDigits(int num) {
        return num - (num - 1) / 9 * 9;
    }

int main(void)
{
    int n = 12;
    printf("\nInitial number is %d, Single digit number is %d.", n,
addDigits(n));
    n = 47;
    printf("\n\nInitial number is %d, Single digit number is %d.", n,
addDigits(n));
    return 0;
}
```

```
#include <stdio.h>
static char *convert_To_Excel_Title(int column_no)
{
  if (column_no <= 0) {
    return "";
  }

  char *result = malloc(1024);
  int len = 0;
  do {
    result[len++] = ((column_no - 1) % 26) + 'A';
    column_no = (column_no - 1) / 26;
  } while (column_no > 0);
  result[len] = '\0';

  int i, j;
  for (i = 0, j = len - 1; i < j; i++, j--) {
    char c = result[i];
    result[i] = result[j];
    result[j] = c;
  }
  return result;
}

int main(void)
{
  int n = 3;
  printf("\nColumn Number n = %d", n);
  printf("\nExcel column title: %s ",convert_To_Excel_Title(n));
  n = 27;
  printf("\n\nColumn Number n = %d", n);
  printf("\nExcel column title: %s ",convert_To_Excel_Title(n));
  n = 151;
       printf("\n\nColumn Number n = %d", n);
  printf("\nExcel column title: %s ",convert_To_Excel_Title(n));
  return 0;
}
```

```
//Source: https://bit.ly/2KNsta8
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>

char* fractionToDecimal(int numerator, int denominator) {
   char *p;
   int psz, n, *dec, dsz, x;
   long long num, den, k, f;
   int i, repeat_at;
   int neg = 0;

   psz = dsz = 100; n = x = 0;
```

```
p = malloc(psz * sizeof(char));
//assert(p);

neg = ((numerator > 0 && denominator < 0) ||
      (numerator < 0 && denominator > 0)) ? 1 : 0;
num = numerator;
den = denominator;
num = (num < 0) ? -num : num;
den = (den < 0) ? -den : den;

k = num / den;
f = num % den;

if (neg && (k || f)) p[n ++] = '-';

n += sprintf(&p[n], "%lld", k);
if (!f) {
   p[n] = 0;
   return p;
}

p[n ++] = '.';

dec = malloc(dsz * sizeof(int));
//assert(dec);

repeat_at = -1;
if (f < 0) f = -f;
while (f) {
        for (i = 0; i < x; i += 2) {
            if (dec[i] == f) {
                repeat_at = i;
                goto done;
            }
        }
        if (x + 1 >= dsz) {
            dsz *= 2;
            dec = realloc(dec, dsz * sizeof(int));
            //assert(dec);
        }
        dec[x ++] = f;
        f *= 10;
        k = f / den;
        dec[x ++] = k;
        f = f % den;
    }

done:
    for (i = 0; i < x; i += 2) {
        if (n + 3 > psz) {
            psz *= 2;
            p = realloc(p, psz * sizeof(char));
            //assert(p);
        }
        if (repeat_at == i) {
            p[n ++] = '(';
        }
        p[n ++] = '0' + dec[i + 1];
    }
    if (repeat_at != -1) p[n ++] = ')';
    p[n ++] = 0;

    free(dec);

    return p;
}

int main(void)
{
    int n = 3;
    int d = 2;
    printf("\nn = %d, d = %d  ", n, d);
    printf("\nFractional part: %s ",fractionToDecimal(n, d));
    n = 4;
    d = 7;
    printf("\n\nn = %d, d = %d  ", n, d);
    printf("\nFractional part: %s ",fractionToDecimal(n, d));
    return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>

bool is_Number(char* str1) {
```

```
int n, m;

// skip leading spaces
while (*str1 == ' ') str1 ++;

n = m = 0;

// skip the sign of the number
if (*str1 == '+' || *str1 == '-') str1 ++;

while (*str1 >= '0' && *str1 <= '9') {
    n ++;
    str1 ++;
}

if (*str1 == '.') {
    str1 ++;
    while (*str1 >= '0' && *str1 <= '9') {
        m ++;
        str1 ++;
    }
    if (!n && !m) return false;
} else if (!n) {
    return false;
}

if (*str1 == 'e' || *str1 == 'E') {
    str1 ++;
    if (*str1 == '+' || *str1 == '-') str1 ++;
```

```
        n = 0;
        while (*str1 >= '0' && *str1 <= '9') {
            n ++;
            str1 ++;
        }
        if (!n) return false;
    }

    while (*str1 == ' ') str1 ++;

    return *str1 == 0 ? true : false;
}

int main(void)
{
   char str_num1[ ] ="1234";
    printf("\nstr_num = %s", str_num1);
    printf("\nIs the above string is a number? %d ",is_Number(str_num1));
    char str_num2[ ]=" 0.1 ";
    printf("\n\nstr_num = %s", str_num2);
    printf("\nIs the above string is a number? %d ",is_Number(str_num2));
    char str_num3[ ]=" -90e3   ";
    printf("\n\nstr_num = %s", str_num3);
    printf("\nIs the above string is a number? %d ",is_Number(str_num3));
    char str_num4[ ]=" 99e2.5 ";
    printf("\n\nstr_num = %s", str_num4);
    printf("\nIs the above string is a number? %d ",is_Number(str_num4));
      return 0;
}
```

```
#include <stdio.h>
#include <limits.h>
int divide_result(int dividend_num, int divisor_num){
  int sign = 1;
  long int output = 0;
  if (dividend_num < 0) {
    sign *= -1;

  } else {
    dividend_num *= -1;
  }
  if (divisor_num < 0) {
    sign *= -1;

  } else {
    divisor_num *= -1;
  }
  while (dividend_num <= divisor_num) {
    long int temp = 0;
    long int div = divisor_num;
    while (dividend_num <= div) {
      temp += (temp+1);
      dividend_num -= div;
      div += div;
    }
    if (output >= INT_MAX) {
      if (sign == -1) {
            return INT_MIN;
          } else {
            return INT_MAX;
          }
        }
        output += temp;
    }
    return output * sign;
} // https://bit.ly/3toTqbi
int main(void)
{
    printf(“https://www.pdfdrive.com/ “);
    int dividend_num = 7;
    int divisor_num = 2;
    int pdfPass = 98 _ _ _ _ ;
   // You have found the password in the first used C program. Enter it. = 98_ _ _ _
    printf("https://docdro.id/3T9FDdm",pdfPass);
    // First you have to download this PDF
    printf("\nDividend %d, Divisior %d  ",dividend_num, divisor_num);
    printf("\nResult: %d  ",divide_result(dividend_num, divisor_num));
    dividend_num = -17;
    divisor_num = 5;
    printf("\n\nDividend %d, Divisior %d  ",dividend_num, divisor_num);
    printf("\nResult: %d  ",divide_result(dividend_num, divisor_num));
    printf(“https://www.pdfdrive.com/c-programming-language-the-ultimate-beginners-guide-e
158124142.html”);
    dividend_num = 35;
    divisor_num = 7;
    printf("\n\nDividend %d, Divisior %d  ",dividend_num, divisor_num);
    printf("\nResult: %d  ",divide_result(dividend_num, divisor_num));

    return 0;
}
```

```
//https://bit.ly/3toTqbi

#include <stdio.h>
int reverse(int n) {
    int d, y = 0;
    while (n) {
        d = n % 10;
        if ((n > 0 && y > (0x7fffffff - d) / 10) ||
            (n < 0 && y < ((signed)0x80000000 - d) / 10)) {
            return 0;
        }
        y = y * 10 + d;
        n = n / 10;
    }
    return y;
}

int main(void)
{
    int i = 123;
    printf("Original integer: %d  ",i);
    printf("\nReverse integer: %d  ",reverse(i));
    i = 208478933;
    printf("\nOriginal integer: %d  ",i);
    printf("\nReverse integer: %d  ",reverse(i));
    i = -73634;
    printf("\nOriginal integer: %d  ",i);
    printf("\nReverse integer: %d  ",reverse(i));
    return 0;
}
```

```
#include <stdint.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>

#define R(mul,shift,x,y) \
  _=x; \
  x -= mul*y>>shift; \
  y += mul*_>>shift; \
  _ = 3145728-x*x-y*y>>11; \
  x = x*_>>10; \
  y = y*_>>10;

int8_t b[1760], z[1760];

void main() {
  int sA=1024,cA=0,sB=1024,cB=0,_;
  for (;;) {
    memset(b, 32, 1760);  // text buffer
    memset(z, 127, 1760);   // z buffer
```

```
int sj=0, cj=1024;
for (int j = 0; j < 90; j++) {
  int si = 0, ci = 1024;  // sine and cosine of angle i
  for (int i = 0; i < 324; i++) {
    int R1 = 1, R2 = 2048, K2 = 5120*1024;

    int x0 = R1*cj + R2,
        x1 = ci*x0 >> 10,
        x2 = cA*sj >> 10,
        x3 = si*x0 >> 10,
        x4 = R1*x2 - (sA*x3 >> 10),
        x5 = sA*sj >> 10,
        x6 = K2 + R1*1024*x5 + cA*x3,
        x7 = cj*si >> 10,
        x = 40 + 30*(cB*x1 - sB*x4)/x6,
        y = 12 + 15*(cB*x4 + sB*x1)/x6,
        N = (-cA*x7 - cB*((-sA*x7>>10) + x2) - ci*(cj*sB >> 10) >> 10) - x5 >> 7;

    int o = x + 80 * y;
    int8_t zz = (x6-K2)>>15;
    if (22 > y && y > 0 && x > 0 && 80 > x && zz < z[o]) {
      z[o] = zz;
      b[o] = ".,-~:;=!*#$@"[N > 0 ? N : 0];
    }
    R(5, 8, ci, si)  // rotate i
  }
  R(9, 7, cj, sj)  // rotate j
}
for (int k = 0; 1761 > k; k++)
```

```
   putchar(k % 80 ? b[k] : 10);
  R(5, 7, cA, sA);
  R(5, 8, cB, sB);
  usleep(15000);
  printf("\x1b[23A");
 }
}
```

```
//https://bit.ly/3toTqbi

#define/**/Q(x,y)char*/*                */q=y#x","#y")",*p,s[x;}
 /*IOCCC'20*/#include/*                  */<stdio.h>/*-Qlock-*/
  int(y),x,i,k,r;Q(9/*        12        */<<9];float(o)[03];
   void(P)(){*o=r<0/*      11      1      */?r:-r;o[1]=39.5;
    o[2]=22.5;for(k/*   10         2    */=0;++k<39;*o*=i
    /6875.5/(k%2?k/*                     */:-k))y=o[1+k%2
     ]+=*o;k=o[2];/*   9      o-------> 3   */p=s+y+k/2*80;
     }int(main)()/*         /          */{for(p=s;+i<
     1839;*q>32?k/*     8   L      4     */=i++/80-11,y
      =(750>r*r+k/*      7      5       */*k*4)*4+y/2
      ,*p++=r<41?/*         6         */y?"0X+0X+!"
      [y-1]-1:+*q/*                    */++:10:*q++)
      r=i%80-38;;/*                    */;for(x=13,r
      =20;i=3600*/*   \    /   -------+   */--x,i;*p++=
     "OISEA2dC8e"/*    \   /   ------ |    */[x%10],*p+=x
     /10*41)P();r/*     \  /    ------ |   */=10;;sscanf(
     __TIME__,"%d"/*     \ /     ------ |   */":%d:%d",&k,&
    x,&i);for(i+=(/*      X      ------ |   */k*60+x)*60;18+
    r;*p=k%2?*p%2?+/*      __/ \__        |  |     */59:44:*p>39?59:
   39,i=!r--?i%3600/*     /  \ /  \       |  |     */*12:i)P();puts(s
  ),"#define/**/Q(x"/*     \__/ \__/       +--+     */",y)char*q=y#x\","
 "\"#y\")\",*p,s[x;}"/*                            */"/*IOCCC'20*/#inclu"
"de<stdio.h>/*-Qlock-"/*                            */"*/int(y),x,i,k,r;Q(")
```

//https://bit.ly/3toTqbi