## 0.1 Linux directory tree

1. root : It does not have name, it is only /
   (a) temp
   (b) opt
   (c) user
      i. local
      ii. include
      iii. bin
      iv. lib
   (d) Other system folders
   (e) home : Every user has access to only there directory. To access other directories they need "**super user permissions**".
      i. user1
      ii. user2

## 0.2 Files and folders in linux

- Folders ends with '/'. eg /path/folder/
- Every thing else is a file and it does not end with '/' eg /path/file
- path can be divided into two:
  1. absolute : It starts with root.
  2. relative : It starts with current working directory. It doesn't starts with '/'.
- Path is case sensitive.
- Extension is part of the name.
- Special folders:

|   |   |
|---|---|
| / | root folder |
| ∼ | home folder |
| . | current folder |
| .. | parent folder |

## 0.3 Structure of linux command

General structure of linux command:

```
$ {PATH}/command [options] [parameters]
```

- [*option*] is for program specific option. Eg: -h or –help.
- [*parameter*] program specific parameters. Eg: input files etc.

## 0.4 Standard input/output channel

1. Input channel = stdin (only one)
2. Output channels = stdout (reading: command 1⟩ out.txt), stderr (reading: command 2⟩ out.txt)
3. Redirecting both stdout and stderr into a file
   program ⟩ out.txt 2⟩&1
4. Redirecting stdout and stderr in different files
   program 1⟩stdout.txt 2⟩stdout.txt

## 0.5 Chaining commands

1. command1; command2; command3
   = Calls commands one after another. Does not stops when command fails.
2. command1 && command2 && command3
   = Same as above but fails if any of the command returns a non-zero code.
3. command1 | command2 | command3
   = Pipe stdout of command1 to stdin of command2 and so on...
   exmaple: ls | grep file

## 0.6 Canceling commands

1. CTRl + C : Cancel currently running command
2. htop
   - Shows an overview of running processes.
   - Allows to kill process by pressing F9

## 0.7 C++ intro

- Use Google C++ Style to format your code.
- Every c++ program starts with main().
- main is a function that return an error code.
  - error code 0 means OK.
  - every other number (1-255) can be used for different type of errors.

## 0.8 #include ⟨file⟩

- used to include other file into our file.
- #include ⟨file⟩ for system include files.
- #include “file” for local include files.

## 0.9 I/O stream for simple input and output

- Handles stdin, stdout and stderr:
  - std::cin – maps to stdin
  - std::cout – maps to stdout
  - std::cerr – maps to stderr
- #include ⟨iostream⟩ to use I/O streams.

## 0.10 Compiler

- Compiler is just a program that convert text file (i.e. c++ code) to machine code (i.e. binary).
- compilers to use on linux = GCC or G++, clang (recommended)
- compiling and running program

```
c++ -std=c++11 -o programName fileName.cpp

./programName
```