**COMP2113 Exam Cheat Sheet**

*Module 1 Linux Environment*

ls, cd ~username, man **x**, mkdir, rmdir, cp, rm -rf, mv, touch, cat, vi, cat >

chmod: rwx, ugoa, +-=

diff: **a**dd a line after line **0** of *fileA*, the line to be added is line **1** of *fileB*; **c**hange line **2,3** of *fileA* to line **3** of *fileB*; **d**elete line **1** from *fileB*, and the files will then be in sync starting at line.

cut -d-f, wc (-line, -word, -byte), sort -n-r-k-t, uniq, spell, su, yum, man cmd

find: [path][-name][-type f/d]

grep: .^$ ?+\* [] \ (){n,m} -E‘ ’

Redirection&Pipe

>: send into a file >>: append

>|: force to be written, 2>&1

0: input 1: output 2: error output

<: Input |: output 🡪 Programm

*Module 2 Shell Script*

#!/bin/bash

chmod u+x [program]

./[program]

variable: read $ \`command`

count: ${#a}

part: ${a:pos:len}

substitute: ${a/from/to}

let: math operation

script\_name: $0 $#

if-else-fi: exist, success, error

compare: string, check, number

for loop

delete: >/dev/null

replace: echo “$0: Oops”>&2

*Module 3 C++*char, int, double, bool

const

cout << “ “ << endl

cin >> ;

strlen()

if ( ) {;} else {;} / condition ? expr1 : expr2

else if

switch (control) { case 1: statement; break; default: statement; } / {case 0: case 1: case 3: statement; break;

while (condition) {statement; }

loop variable: to count the no. of iterations

initialization: of loop variable

condition: for continuation

updating of loop variable inside the loop body

Sentinel-controlled, Counter-controlled, Flag-controlled

for (initialization; condition; updating) {statement\_1;

...; statement\_n;}

break, continue

*Module 4 makefile*

Header, Function declare, main, definition

// gcd.h

#ifndef GCD\_H

#define GCD\_H

int gcd(int a, int b);

#endif

$ g++ -pedantic-errors -std=c++11 gcd\_main.cpp gcd.cpp -o gcd

makefile:

gcd.o: gcd.cpp gcd.h

g++ -c gcd.cpp

gcd\_main.o: gcd\_main.cpp gcd.h g++ -c gcd\_main.cpp

gcd\_main: gcd.o gcd\_main.o

g++ gcd.o gcd\_main.o -o gcd\_main

use variables:

$() $@: target $^: dependency $<: first dependency

clean:

rm -f gcd\_main gcd.o gcd\_main.o gcd.tgz

tar:

tar -czvf gcd.tgz \*.cpp \*.h

.PHONY: clean tar

*Module 5 Funct & Recurs*

divide and conquer

predefined function:

<iostream> cin cout

<cmath> sqrt pow

<cstdlib> rand srand(time(NULL))

*Q.2 how to generate a random integer in the range of [0, 100]?*

*Answer:*

*rand() % 101*

<ctime> time(NULL)

function definition:

type\_ret func\_name(type1 par1, type2 par2, ...) {

// variable declarations ...

// executable statements ...

}

void function: return nothing

function call

function declaration:

definition before the call

or

with declaration

type\_ret func\_name(type1 par1, type2 par2, ...); or

type\_ret func\_name(type1, type2, ...);

Global Variable: constant

pass-by-reference: &x to change together, use void