**grep**:-v:outputNotMatch;-c:NumberOccur;-l:OutputMatchLineOnlyForMuilt-file;,-r:searchRecurisivelyInFile**Regex**:\w(A-Za-z0-9),\W(!\w),\a(A-Za-z),\<&\>(begin/end string),\d(0-9),\D(!\d),\s(“ ”\t),\S(!\s),\u(A-Z),\l(a-z),\x(A-Fa-f0-9),\b(zero-width bounty) ?:0-1char before +:1+char before |:choose char before/after .:one char []:char range [^]:case not match inside brackets {m,n}:m-n time ():small expression use later with $n ^:start $:end **LAYER of abstract machine**: user>user program>system utilities>kernel>microkernel>hardware

**SystemFoundations**:files,device\_interface(network/storage…),utilities&utility\_process,kernel/microkernel,command(method to get user intention & process input to output),process(computational entity/program in execution) **AbstractionsInUNIX**:everythingIsFile(SequenceOfBytes)/DefaultFileFormat:ASCII/command=buildInOperation&Program/allConnected

**ConfigurationFiles(SetUp)**:*.bashrc*:bash;*.profile*:sh&bash&kshLogin;*.tcshrc*:tcsh;*.cshrc*:csh&tcsh;*.login*:csh&tcshLogin**ImportantDirectories:*/*:**root;*./*:current;*~/*:home;*~user/*:specificUser;..:Parent **Man:**1.GeneralCommands 2. SystemCalls 3.CLibraryFunctions4.SpecialFiles&Driver5.FileFormats&Conventions6.Games&Screensavers7.Miscellanea8.SystemAdministrationCommands&Daemons **Process:** Foreground: Associated With Input & Shell Wait Complete. Background: Executed When Permitted & Shell Not Wait Complete. Suspended: Inactive Background Process(may Active Later) Job: Suspended Background Process. **Shell Char**:/s Delimiter;\* FilenameWildcrad;$ nextTokenIsVar;\ preventFunctionOfNextMetaCharacter; ‘’(prevent$work)&””quote **Shell Programmming:** start:”#!/(path to interpreter)”comment”#...”run:sh file;./file;sh < file **ShellScriptVariables(start$)**: ?: exitStatus; $: PID of Process; 1-n: parameter1-n; @ : allParameter; #: NumberOfParameter 0: nameOfProgram (**Quoting**: Same As Shell Char Part.) (**Self-definedVariables** MustBeAllCapitalLetter&SeperateNameWithEnvVar) (CanModifyEnvVar&UsedOutSideShellScript) **Shift(Command):**moveVarValueToOneBefore **DotFile(AScriptFile):**source+dotFileName make it use interactively by shell(equal to ./File) **ShellDebugging:** set –v(verbose): provide additional detail to what computer is doing ; set-x(execution): show what command is and it`s output next line. **exprCommand:** (A|B:returnAIfAisNotNull/0;ElesB) (A&B:returnAIfA&BneitherNUll/0;Else0) (match STRING REGEXP:STRINGMatchRegexReturn1;Else0) (substr STRING n y:SubStrOfSTRINGStrartBY n & have length y) (index STRING CHARS: return pos of CHARS in STRING) (length STRING: return lengthOfSTRING) **exit value**:0->if expression is neither null nor 0 1-> if expression is null or 0 2 -> if expression is syntactically invalid. **ExitCommand**: End process and return Exit value. **CaseControl/Modularization:Case:(**if;then;elif;then;else;fi)(while condition do … done)(for var(i) in list($@) do done)**Modularization**:function(){…} **Operators**:**Reational(number value)**:-eq(=);-ne(!=);-gt(>);-lt(<);-ge(≥);-le(≤);(**String**:=;!=;-z(size=0);-n(size>0); str(not empty string));(**File:**-e(file exist);-d(is directory)) **AWK**: read data as string/float;process text one record(default:line) each time; interpreted language(don`t need compile)**Run**:1.awk ‘script’(infile);2.awk -f scriptfile (infile);3. awk ‘condition{action}’ [filename];( $0 means everything in record;$1-n 1-n`s field in file)4.statements:if-else;while-do;for;do-while;continue;printf; **build-in variables**: NR: number of records input thus far; NF: # of fields in current record; FILENAME: name of current input file; FS: field separator characters(default any-whitespace; can set to regex); RS: record separator(default /n; can set to regex); OFS: output field separator; ORS: output record separator; string printed at end of print; OFMT: format code for numeric output.; variables can be set within script or on invocation of awk with -v var=val; **selector**: (!)/regex/ : (false)true if the record matches regex; $N~(!)/regex/ : (false)true if field N matches regex; $N op value: op in { == , !=, <, >, <=, >= };value is a string or number; boolean ops: &&, ||, !; **make it a shell command:** #!/bin/awk -f/chmod u+x

**Memory**:addressedArrayOfBytes;changeSizeAnsShapeBaseOnRequire;**Content**:data/addressToAnother/undefined/ProgrammeInstruct;**ConstructVar**:rceiveMemorySpace,LocalNameAsPointer,AlwaysInternalize;\*&foraddress**.sizeOf()**returnAIntByteSize;**LongInt&pointerTo**:32-4/64-8;Use<inttypes.h>keep**SizeOfIntFix**

**Commands**:/**bin**:minimal set command /**usr**/**bin**:most user commands /**usr**/**local**/**bin**:local versions of commands **Help** **with** **command** 1.man command 2.apropos 3.whatis 4.info command 5.khelpcenter

Program execution envirnment: **HOME**:your home directory**PATH**:directories that are automatically searched**TERM**: yout output(display) device Variable **SHELLOPTS** shows a list of enabled shell options

**chmod**: 0 No permission(---)1Execute permission(--x) 2Write permission(-w-) 4Read permission(r--) **du**-estimate file space usage **df**-display the amount of available disk space for file systems on which the invoking user has appropriate read access **tee-**outputObjectBothToTerminal&ObjectFile **whoami** --- returns your username **kill *PID*** --- kills (ends) the processes with the ID you gave **nice** (/usr/bin/nice)-run a program with modified scheduling priority **time**(/usr/bin/time**)**-time a simple command or give resource usage **make** - GNU make utility to maintain groups of programs **ln** -**s** <filename> <link> creates symbolic link to file **tar** <operation> [options] -f, --file x –extract **uname** -**a** shows kernel information

**I/O redirection command:** 0 stdin; 1 stdout; 2 stderr **avoid** **overwrite** in **redirection**:**set –C** or **set –o noclobber** (override with >|) use **>&** in bash to redirect operators **2>&1** = add stderr to the stdout

^**C** end-of-input ^**D** end-of –process **!!:** repeat the last command used !**number**: repeat last number !**string**: repeat last string **env**/**printenv**/**set**:print all environment variable **shortcut of command: alias** Var =”command” see all alias by **alias** remove by **unalias** change file between OS:**unix2dos / dos2unix** **tr ‘\r’ ‘\n’ mtools** on tuxworld **chgrp**:change group ownership **chown**:chown owner of file

file command: determine file type of a ordinary file---Pattern matching file name: **\*** any number of any character **?** any one character **[]** a set of character **stty** - change and print terminal line settings

**type+command**: show command if it is a builtin command

**sed:** stream editor for filtering and transforming text

**ldconfig**命令的用途主要是在默认搜寻目录/lib和/usr/lib以及动态库配置文件/etc/ld.so.conf内所列的目录下，搜索出可共享的动态链接库（格式如lib\*.so\*）,进而创建出动态装入程序(ld.so)所需的连接和缓存文件。  
**Characteristics of Good Style** clear and simple • straightforward logic • conventional (programming) language use • natural expression • meaningful names • neat formatting • helpful comments

**uniq** - report or omit repeated lines

**Programming style**:**Names**: **1.**informative, concise, memorable, and pronounceable **2**.broader the scope of variable, more informative the name should be**3**.be consistent**4**.be accurate**5**.active name for function**Expression and Statements**:**1**.write in the way makes their meaning clear2.clearest code3.use natural form for expressions4.add words to resolve misunderstanding5.break up complex expressions6.clear7.careful about side-effect **Indentation and Dspacing 1.**space to suggest grouping **2**.indent缩进shows structure of function **3.** sprawling layouts detract fromreadability **4.**constant indention/brace style**Idioms** conventional way that experienced programmer choose to style codeeasy to recognize, understandeasy to writevariation cause genuine difference

**Testing**: testing is a determined, systematic attempt to break a program that you think is working

**Degrees,** based on difficulty of input : 0. none 1. very trivial, "program compiles" 2. trivial, no input • e.g. "I can start the program and it doesn't crash" 3.all input for level 2, plus basic input  4.all input for level 3, plus reasonable, expected input  5.all input for level 4, plus difficult input (e.g. exercise boundary conditions)  6.all input for level 5, plus unreasonable, illegal input  7.all possible input

**path** coverage: the path of an input will travel inside the function **branch** coverage:if all branch(if/else statement) is covered?**condition** coverage:if any condition is tested with test cases?

integration test: test combined modules(模块) and interface between groups **Regression Testing** sequence of tests that compare results from a new version with that from the previous version "don't break something that was working before"important both during incremental development and during maintenance implicit assumption that the previous version computes the correct answer **test** **scaffold**: create a simple Structure for function testing **Systematic Testing:** test incrementally write part test it if pass; write more and testke repeat integrate as components are completed test simple part first test minimum needs of another program you are not test onExcept what happen if pass or if not;Keep recordSVN, readme file, commentsIf program handed an opposite-side function, check both sidesPreserve(保护)input propertyTest scaffold: create a simple Structure for function testing

**Performance&Profiling:** **gprof**(1)example compile and link with -pg option note documentation within gprof output

**OptimizationAvenues:**contribution from compiler optimization **O, -O1**: compiler tries to reduce code size and execution time, without performing any optimizations that take a great deal of compilation time **O2**: optimize even more, taking more compilation time **O3**: optimize yet more, taking even more compilation time

**Portability: little (low) endian**: low-order byte stored at lower address **big (high) endian**: low-order byte stored at bigger (higher) address  **little endian**=LSB("least significant byte")  **big endian =** MSB ("most significant byte")  **valgrind** on LINUX/UNIX originally a memory-leak detector. Now has many additional profiling capabilities consists of a simulated CPU, and a series of debugging and profiling tools

**sample**(1) on BSD systems stops the indicated process at user-defined intervals. records the current function being executed, and how the function was called; i.e. examine call stack then lets the application continue produces a call tree

**Debugging**:**Bohr** **bug**: a repeatable bug; one that manifests reliably under a possibly unknown but well-defined set of conditions **Heisenbug**: a bug that disappears or alters its behaviour when one attempts to probe or isolate it**Common Types of Bugs:** 1.syntax bugs2.Arithmetic bugs(div by 0)3.logic bugs(infinite loops infinite recursion off-by-one error)4.resource bugs(null pointer dereference using uninitialized variable didn't free/delete all dynamically allocated memory - memory leak re-use of already-freed memory confusing statically-allocated, local, and dynamically- allocated memory)5.side-effect bugs

**tools:** -syntax-directed editor compiler options –Wall –Wextra assertion **DEBUG flag**(sort through 2 much output) **wolf-fences(**insert printf**)**

**Debugging Techniques:** 1.DEBUG flag and printf() statements 2.logging

**Svn**

**Centralized VCS**  2Use copy-modify-merge **Repository** •the single copy of master sources •file system tree **working copy •**ordinary directory tree on local system •checkout-“pull”

•atomic commit: all or none happened, retain this atomicity in carshes, problems **update(to see change):** cause a read ”pull” from repository **version numbers:**(comiitted reversion 1)

**trunk:** main development sequence **branch:** copy of current state of trunk(svn copy)

**•svn merge:** apply difference b/t 2 sources to working copy path(merge branch back to trunk)

**tag:** a snapshot(V1.0), can be copy of trunk or branch

*//if [ ! -d $SvnRepos ] ; then svnadmin create $SvnRepos ; fi //svn --username=NSID //svn checkout –r 3(ask for version 3) //svnadmin recover(lose data) //svn merge –reintegrate $SvnURL //Wrk=`pwd`/example2Wrk if [ ! -d $Wrk ] ; then mkdir $Wrk ; fi //pushd $Wrk*

*//svn checkout $SvnTrunk(now working cp) //pushd trunk //svn import . $SvnURL (import everything) //svn status* ***–u****:show-updates, add working reversion* ***–v****:verbose, print reversion every item* ***–q****:--quiet(少)printlocally modified items //svn copy $SvnBranches/trivial\_in $SvnTags/V1.0*

**Libraries**

**Linker:** combines .o files static: inside executable dynamic:in shared library(save RAM usage) **2.-L:** local object modules –L/student/214/lib **3reentrant:** 无static non-constant data 无return address to static 无own code只 work on parameter **4PIC:** 为了library fully shared(a body of machine code)*//gcc –Fpic* **5ldd(1)** examine the shared libraries that queens will make use of -I/student/214/include **(**search /student/214/include for **“system include files” )**g++ -o queens queens.o -L/student/214/lib –lqueens**(ld looks for libraries, it is also to search the directory /student/214/lib)**

*//gcc -Wall -Wextra -c -o example8a.o example8a.c //ar rcs libexample8.a example8a.o* ***# r*** *- insert into the library with replacement; in a new archive could have also used q****# c*** *- create a new archive* ***# s*** *- write an object index into the archive; means we don't have to,**run ranlib afterward* ***# q*** *- quick append (add file to end without checking for replacement)****# t*** *- display a table listing the contents of the archive*

*// ar t libexample8.a*

***//****install --group=cmpt214 --mode=664 libexample8.a /student/214/lib(linker 放入 /lib)*

*//gcc -static -o example8-s example8.o -L/student/214/lib -lexample8 (Static linked)*

*//gcc -o example8-h example8.o -L/student/214/lib -lexample8 (Dynamically linked)*

**makefile**

**nice:** control “nice number” (/usr/bin/nice) **time:** real:1S user:1.1s sys:0.1s (/usr/bin/time)

**motivation:** Desirable: only compile what changed/ light-weight mechanism **make:** target(must up-to-date, any object in system) +dependency(list of targets that influence the main target) +action (commands that exexuted if dependency object has a time-stamp more recent than target) +rule(specification of 前面的3个)

**Dependency graph** .**PHONY** tells make that the rule doesn't generate a file with the target name (eg. .PHONY: all tests clean**) - CC :** C compiler to use**- CXX :** C++ compiler to use

**- CFLAGS :** flags to C compiler**- CXXFLAGS:** flags to C++ compiler - LDFLAGS: flags to linker (ld)**Implicit rule:** (不写compiler)**-n** particularly useful**-W *file*** : pretend that target *file* has just been modified**-trace (on LINUX version)** useful for learning about make

**Iterative Software Development**

**1.**decompose large problem into a small problem plus series of increments **2.main step:** initialization(initial prototype)+iterative step(additional function from plan, user feedback analysis)+project control(modify plan in analysis results)**3.Prototyping:** “quick & dirty” + feedback & refine

**TDD**

**Test first**: Do not write a line of code until you have written a failing unit test •Do not write more of a unit test than is necessary to fail. •Do not write more code than is necessary to pass the failed test. **Coding •**write unit tests for next function •code initially fails the tests •write code that makes the unit tests pass; iterate with new functions/tests •more and more of the code works correctly **Completed Program •** run the completed program, showing that all of the end-to-end tests pass

grep:-v:outputNotMatch;-c:NumberOccur;-l:OutputMatchLineOnlyForMuilt-file;,-r:searchRecurisivelyInFileRegex:\w(A-Za-z0-9),\W(!\w),\a(A-Za-z),\<&\>(begin/end string),\d(0-9),\D(!\d),\s(“ ”\t),\S(!\s),\u(A-Z),\l(a-z),\x(A-Fa-f0-9),\b(zero-width bounty) ?:0-1char before +:1+char before |:choose char before/after .:one char []:char range [^]:case not match inside brackets {m,n}:m-n time ():small expression use later with $n ^:start $:end UNIXTwoMainBranch:BSD/System3&5 layer of abstract machine: user>user program>systemutilities>kernel>microkernel> hardware SystemFoundations:files,device\_interface(network/storage…),utilities&utility\_process,kernel/microkernel,command(method to get user intention & process input to output),process(computational entity/program in execution) AbstractionsInUNIX:everythingIsFile(SequenceOfBytes)/DefaultFileFormat:ASCII/command=buildInOperation&Program/allConnected

ConfigurationFiles(SetUp):.bashrc:bash;.profile:sh&bash&kshLogin;.tcshrc:tcsh;.cshrc:csh&tcsh;.login:csh&tcshLoginImportantDirectories:/:root;./:current;~/:home;~user/:specificUser;..:Parent Man:1.GeneralCommands 2. SystemCalls 3.CLibraryFunctions4.SpecialFiles&Driver5.FileFormats&Conventions6.Games&Screensavers7.Miscellanea8.SystemAdministrationCommands&Daemons Process: Foreground: Associated With Input & Shell Wait Complete. Background: Executed When Permitted & Shell Not Wait Complete. Suspended: Inactive Background Process(may Active Later) Job: Suspended Background Process. Shell Char:\s Delimiter;\* FilenameWildcrad;$ nextTokenIsVar;\ preventFunctionOfNextMetaCharacter; ‘’(prevent$work)&””(preventMostMetaChar)quote Shell Programmming: start:”#!/(path to interpreter)”comment”#...”run:sh file;./file;sh < file ShellScriptVariables(start$): ?: exitStatus; $: PID of Process; 1-n: parameter1-n; @ : allParameter; #: NumberOfParameter 0: nameOfProgram (Quoting: Same As Shell Char Part.) (Self-definedVariables MustBeAllCapitalLetter&SeperateNameWithEnvVar) (CanModifyEnvVar&UsedOutSideShellScript) Shift(Command):moveVarValueToOneBefore DotFile(AScriptFile):source+dotFileName make it use interactively by shell(equal to . File) ShellDebugging: set –v(verbose): provide additional detail to what computer is doing ; set-x(execution): show what command is and it`s output next line. exprCommand: (A|B:returnAIfAisNotNull/0;ElesB) (A&B:returnAIfA&BneitherNUll/0;Else0) (match STRING REGEXP:STRINGMatchRegexReturn1;Else0) (substr STRING n y:SubStrOfSTRINGStrartBY n & have length y) (index STRING CHARS: return pos of CHARS in STRING) (length STRING: return lengthOfSTRING) exit value:0->if expression is neither null nor 0 1-> if expression is null or 0 2 -> if expression is syntactically invalid. ExitCommand: End process and return Exit value. CaseControl/Modularization:Case(ConditionalCommands):(if;then;elif;then;else;fi)(while condition do … done)(for var(i) in list($@) do done)Modularization:function(){…} Operators:Reational(number value):-eq(=);-ne(!=);-gt(>);-lt(<);-ge(≥);-le(≤);(String:=;!=;-z(size=0);-n(size>0); str(not empty string));(File:-e(file exist);-d(is directory)) AWK: read data as string/float;process text one record(default:line) each time; interpreted language(don`t need compile)Run:1.awk ‘script’(infile);2.awk -f scriptfile (infile);3. awk ‘condition{action}’ [filename];( $0 means everything in record;$1-n 1-n`s field in file)4.statements:if-else;while-do;for;do-while;continue;printf; build-in variables: NR: number of records input thus far; NF: # of fields in current record; FILENAME: name of current input file; FS: field separator characters(default any-whitespace; can set to regex)SetBy -F; RS: record separator(default /n; can set to regex); OFS: output field separator; ORS: output record separator; string printed at end of print; OFMT: format code for numeric output.; variables can be set within script or on invocation of awk with -v var=val; selector: (!)/regex/ : (false)true if the record matches regex; $N~(!)/regex/ : (false)true if field N matches regex; $N op value: op in { == , !=, <, >, <=, >= };value is a string or number; boolean ops: &&, ||, !; make it a shell command: #!/bin/awk -f/chmod u+x

Memory:addressedArrayOfBytes;changeSizeAnsShapeBaseOnRequire;Content:data/addressToAnother/undefined/ProgrammeInstruct;ConstructVar:rceiveMemorySpace,LocalNameAsPointer,AlwaysInternalize;\*&foraddress.sizeOf()returnAIntByteSize; LongInt&pointerTo:32-4bytes/64-8bytes;Use<inttypes.h>keepSizeOfIntFix FileNameWildCardExpansion:\*anychar;?anyOneChar;{1,..}optionGroup FileNameCompletion:\t

/bin:minimal set command /usr/bin:most user commands /usr/local/bin:local versions of commands Search command 1.man command 2.apropos 3.whatis 4.info command 5.khelpcenter 6. --help

Program execution envirnment: HOME:your home directoryPATH:directories that are automatically searchedTERM: yout output(display) device Variable SHELLOPTS shows a list of enabled shell options

chmod: 0 No permission(---)1Execute permission(--x) 2Write permission(-w-) 4Read permission(r--) du-estimate file space usage df-display the amount of available disk space for file systems on which the invoking user has appropriate read access **commands**: **tee**-read output of command A and write to standard output and CommandB/processTwoCommandTogether

whoami --- returns your username kill PID --- kills (ends) the processes with the ID you gave nice (/usr/bin/nice)-run a program with modified scheduling priority time(/usr/bin/time)-time a simple command or give resource usage make - GNU make utility to maintain groups of programs ln -s <filename> <link> creates symbolic link to file tar <operation> [options] -f, --file x –extract c--create uname -a shows kernel information bg:run Jobs In Background pr printFile -n LineNumber -t removeDefaultEmptyLineBypr--version:version number of commandgcc: -c make .o type-file -g provideDebuggingForGDB(core file)touch a{1..2}.txtls a?\*{a,b,c}start with a and end with a/b/c cmp - compare two files byte by byte stty - change and print terminal line settings export set a user defined var as a EnvVar;shellBuiltIn:.:;[;alias;bg;bind;break;builtin;caller;cd;command;compgen;complete;compopt;continue;declare;dirs;disown;echo;enable;eval;exec;exit;export;false;fc;fg;getopts;hash;help;history;jobs;kill;let;local;logout;mapfile;popd;printf;pushd;pwd;read;readarray;readonly;return;set;shift;shopt;source;suspend;test;times;trap;true;type;typeset;ulimit;umask;unalias;unset;wait;FileNamePattern \*0-n;?1;[setOffChoose]

I/O redirection command: 0 stdin; 1 stdout; 2 stderr avoid overwrite in redirection:set –C or set –o noclobber (override with >|) use >& in bash to redirect operators 2>&1 = add stderr to the stdout

CTRL+A move to beginning of line B moves backward one character C halts the current command D deletes one character backward or logs out of current session, similar to exit E moves to end of line F moves forward one character G aborts the current editing command and ring the terminal bell J same as RETURN K deletes (kill) forward to end of line L clears screen and redisplay the line M same as RETURN N next line in command history O same as RETURN, then displays next line in history file CTRL+P previous line in command history R searches backward S searches forward T transposes two characters U kills backward from point to the beginning of line V makes the next character typed verbatim W kills the word behind the cursor X lists the possible filename completefions of the current word Y retrieves (yank) last item killed Z stops the current command, resume with fg in the foreground or bg in the background^C end-of-input ^D end-of –process !!: repeat the last command used !number: repeat last number !string: repeat last string env/printenv/set:print all environment variable shortcut of command: alias Var =”command” see all alias by alias remove by unalias change file between OS:unix2dos / dos2unix tr ‘\r’ ‘\n’ mtools on tuxworld chgrp:change group ownership chown:chown owner of file **file command**: determine file type of a ordinary file---Pattern matching file name: \* any number of any character ? any one character [] a set of character stty - change and print terminal line settings **type+command**: show command if it is a builtin command**sed**: stream editor for filtering and transforming textldconfig命令的用途主要是在默认搜寻目录/lib和/usr/lib以及动态库配置文件/etc/ld.so.conf内所列的目录下，搜索出可共享的动态链接库（格式如lib\*.so\*）,进而创建出动态装入程序(ld.so)所需的连接和缓存文件。**uniq** - report or omit repeated lines

Characteristics of Good Style clear and simple • straightforward logic • conventional (programming) language use • natural expression • meaningful names • neat formatting • helpful comments

Programming style:Names: 1.informative, concise, memorable, and pronounceable 2.broader the scope of variable, more informative the name should be3.be consistent4.be accurate5.active name for functionExpression and Statements:1.write in the way makes their meaning clear2.clearest code3.use natural form for expressions4.add words to resolve misunderstanding5.break up complex expressions6.clear7.careful about side-effect Indentation and Dspacing 1.space to suggest grouping 2.indent缩进shows structure of function 3. sprawling layouts detract from readability 4.constant indention/brace styleIdioms conventional way that experienced programmer choose to style codeeasy to recognize, understandeasy to writevariation cause genuine difference

Testing: testing is a determined, systematic attempt to break a program that you think is working.SuccessfulTest:UncoverABug

Degrees, based on difficulty of input : 0. none 1. very trivial, "program compiles" 2. trivial, no input • e.g. "I can start the program and it doesn't crash" 3.all input for level 2, plus basic input  4.all input for level 3, plus reasonable, expected input  5.all input for level 4, plus difficult input (e.g. exercise boundary conditions)  6.all input for level 5, plus unreasonable, illegal input  7.all possible input

path coverage: the path of an input will travel inside the function branch coverage:if all branch(if/else statement) is covered?condition coverage:if any condition is tested with test cases?

integration test: test combined modules(模块) and interface between groups Regression Testing sequence of tests that compare results from a new version with that from the previous version "don't break something that was working before"important both during incremental development and during maintenance implicit assumption that the previous version computes the correct answer test scaffold: create a simple Structure for function testing Systematic Testing: test incrementally write part test it if pass; write more and testke repeat integrate as components are completed test simple part first test minimum needs of another program you are not test on except what happen if pass or if not keep record svn, readme file, comments if program handed an opposite-side function, check both sides preserve(保护)input property test scaffold: create a simple Structure for function testing

Performance&Profiling: gprof(1)example compile and link with -pg option note documentation within gprof output 产生程序运行时某些函数的调用次数、执行时间等等宏观信息（函数执行单间，函数调用关系图） 1.gcc –o test –pg test.c 2../test 3.gprof test gmon.out –b

-b 不再输出统计图表中每个字段的详细描述。-p 只输出函数的调用图（Call graph的那部分信息）。-q 只输出函数的时间消耗列表。-e Name 不再输出函数Name 及其子函数的调用图（除非它们有未被限制的其它父函数）。可以给定多个 -e 标志。一个 -e 标志只能指定一个函数。-E Name 不再输出函数Name 及其子函数的调用图，此标志类似于 -e 标志，但它在总时间和百分比时间的计算中排除了由函数Name 及其子函数所用的时间。-f Name 输出函数Name 及其子函数的调用图。可以指定多个 -f 标志。一个 -f 标志只能指定一个函数。-F Name 输出函数Name 及其子函数的调用图，它类似于 -f 标志，但它在总时间和百分比时间计算中仅使用所打印的例程的时间。可以指定多个 -F 标志。一个 -F 标志只能指定一个函数。-F 标志覆盖 -E 标志。-z 显示使用次数为零的例程（按照调用计数和累积时间计算）**OptimizationAvenues**:contribution from compiler optimization O, -O1: compiler tries to reduce code size and execution time, without performing any optimizations that take a great deal of compilation time O2: optimize even more, taking more compilation time O3: optimize yet more, taking even more compilation time

Set OPT to O1/O2/O3:优化过的代码变短了;1. 清除冗余：比如删除没有使用的变量、无作用的代码之类的。2. 预测结果：直接计算出编译时已知的量（正如上面的）；判断程序数据规模，合理减小内存占用。3. 扩展代码：拆开固定次数的循环、变为顺序结构（执行循环时代码是来回跳跃执行的，不如顺序执行效率高）；内联函数。4. 优化存储：使用CPU寄存器（速度比内存快）来储存频繁操作的变量、数据对齐储存提高读写效率。

Portability: little (low) endian: low-order byte stored at lower address big (high) endian: low-order byte stored at bigger (higher) address  little endian=LSB("least significant byte")  big endian = MSB ("most significant byte")  valgrind on LINUX/UNIX originally a memory-leak detector. Now has many additional profiling capabilities consists of a simulated CPU, and a series of debugging and profiling tools

sample(1) on BSD systems stops the indicated process at user-defined intervals. records the current function being executed, and how the function was called; i.e. examine call stack then lets the application continue produces a call tree

Debugging:Bohr bug: a repeatable bug; one that manifests reliably under a possibly unknown but well-defined set of conditions Heisenbug: a bug that disappears or alters its behaviour when one attempts to probe or isolate it Types of Bugs: 1.syntax bugs2.Arithmetic bugs(div by 0)3.logic bugs(infinite loops infinite recursion off-by-one error)4.resource bugs(null pointer dereference using uninitialized variable didn't free/delete all dynamically allocated memory - memory leak re-use of already-freed memory confusing statically-allocated, local, and dynamically- allocated memory)5.side-effect bugs

tools: -syntax-directed editor compiler options –Wall –Wextra assertion DEBUG flag(sort through 2 much output) wolf-fences(insert printf)

Debugging Techniques: 1.DEBUG flag and printf() statements 2.logging Core file: contain a piece of memory usage when the termination of program(produceBygcc)

SVN:Centralized VCS 2Use copy-modify-merge Repository •the single copy of master sources •file system tree working copy •ordinary directory tree on local system •checkout-“pull”

•atomic commit: all or none happened, retain this atomicity in carshes, problems update(to see change): cause a read ”pull” from repository version numbers:(comiitted reversion 1) trunk: main development sequence branch: copy of current state of trunk(svn copy) svn merge: apply difference b/t 2 sources to working copy path(merge branch back to trunk) tag: a snapshot(V1.0), can be copy of trunk or branch

//if [ ! -d $SvnRepos ] ; then svnadmin create $SvnRepos ; fi //svn --username=NSID //svn checkout –r 3(ask for version 3) //svnadmin recover(lose data) //svn merge –reintegrate $SvnURL //Wrk=`pwd`/example2Wrk if [ ! -d $Wrk ] ; then mkdir $Wrk ; fi //pushd $Wrk

//svn checkout $SvnTrunk(now working cp) //pushd trunk //svn import . $SvnURL (import everything) //svn status –u:show-updates, add working reversion –v:verbose, print reversion every item –q:--quiet(少)printlocally modified items //svn copy $SvnBranches/trivial\_in $SvnTags/V1.0

Libraries: Linker: combines .o files static: inside executable dynamic: in shared library(save RAM usage) 2.-L: local object modules –L/student/214/lib 3reentrant: 无static non-constant data 无return address to static 无own code只 work on parameter 4PIC: 为了library fully shared(a body of machine code) 5ldd(1) examine the shared libraries that queens will make use of -I/student/214/include (search /student/214/include for “system include files” )

g++ -o queens queens.o -L/student/214/lib –lqueens(ld looks for libraries, it is also to search the directory /student/214/lib)

Use -I to link and cretae .o file; use -L & -l for link library when create .out file

//gcc –Fpic//gcc -Wall -Wextra -c -o example8a.o example8a.c //ar rcs libexample8.a example8a.o # r - insert into the library with replacement; in a new archive could have also used q# c - create a new archive # s - write an object index into the archive; means we don't have to, run ranlib afterward # q - quick append (add file to end without checking for replacement)# t - display a table listing the contents of the archive

// ar t libexample8.a (ar create a static library; gcc -shared create a dynamic library)

//install --group=cmpt214 --mode=664 libexample8.a /student/214/lib(linker 放入 /lib)

//gcc -static -o example8-s example8.o -L/student/214/lib -lexample8 (Static linked)

//gcc -o example8-h example8.o -L/student/214/lib -lexample8 (Dynamically linked)

makefile: nice: control “nice number” (/usr/bin/nice) time: real:1S user:1.1s sys:0.1s (/usr/bin/time)

motivation: Desirable: only compile what changed/ light-weight mechanism make: target(must up-to-date, any object in system) +dependency(list of targets that influence the main target) +action (commands that exexuted if dependency object has a time-stamp more recent than target) +rule(specification of 前面的3个) .PHONY tells make that the rule doesn't generate a file with the target name (eg. .PHONY: all tests clean) - CC : C compiler to use- CXX : C++ compiler to use - CFLAGS : flags to C compiler- CXXFLAGS: flags to C++ compiler - LDFLAGS: flags to linker (ld)Implicit rule: (不写compiler)-n particularly useful-W file : pretend that target file has just been modified-trace (on LINUX version) useful for learning about make Iterative Software Development 1.decompose large problem into a small problem plus series of increments 2.main step: initialization(initial prototype)+iterative step(additional function from plan, user feedback analysis)+project control(modify plan in analysis results)3.Prototyping: “quick & dirty” + feedback & refine

TDD: Test first: Do not write a line of code until you have written a failing unit test •Do not write more of a unit test than is necessary to fail. •Do not write more code than is necessary to pass the failed test. Coding •write unit tests for next function •code initially fails the tests •write code that makes the unit tests pass; iterate with new functions/tests •more and more of the code works correctly Completed Program • run the completed program, showing that all of the end-to-end tests pass //what is TDD//common question & concern//introduction to TDD//in C//





