THE HONG KONG UNIVERSITY OF SCIENCE & TECHNOLOGY

COMP111: Software Tools

Final Exam

Spring 1999

31 May 1999

Student Name: ____

Student Number: ____

Lab Section: ____

TA: ____

Instructions:

1. There are 20 problems worth 100 points total.
2. Check that you have all 13 pages.
3. Answer all questions in the space provided, and **circle** your answer. Rough work can be done only on the back pages.
4. Leave all pages stapled together.
5. The examination period will last for **120 minutes.**
6. Stop writing immediately when the time is up.

For Grading Purposes Only:

Page 2:	Problem 1-2	______________	/ 5
Page 3:	Problem 3-4	______________	/ 6
Page 4:	Problem 5	______________	/ 5
Page 5:	Problem 6-8	______________	/ 12
Page 6:	Problem 9-10	______________	/ 6
Page 7:	Problem 11-12	______________	/ 7
Page 8:	Problem 13-14	______________	/ 8
Page 9:	Problem 15-16	______________	/ 9
Page 10:	Problem 17	______________	/ 13
Page 11:	Problem 18	______________	/ 7
Page 12:	Problem 19	______________	/ 14
Page 13:	Problem 20	______________	/ 8
Total:		______________	/100

Array basics:
1) (2 points) What is the value of `$n` after the following Perl statements are executed?

```
$n = 4;
@a = (1,2,3);
for($i=0; $i<=$#a; $i++){
     $n += $a[$i];
}
```

Answer: 10

List Basics:
2) (3 points) What is the value of `@a` after the following Perl statements are executed?

```
@b = (5,6,7);
foreach (reverse @b){
     do{
          push(@a, $_);
          $_--;
     }until($_==4);
}
```

Answer: 7 6 5 6 5 5
1 point: left 7 6 5
1 point: middle 6 5
1 point: right 5

Hash:

3) (3 points) What is the output?

```
     #!/usr/local/bin/perl5 -w
     $n = 2;
     $bill{"Gates"} = "rich and famous";
     $bill{"Mr."} = "too excited";
     $bill{1} = "very rich";
     $bill{$n} = "very excited";
     $bill{3} = 67;
     @a = (3,2,1);
     print "@bill{@a}\n";
```

Answer:

```
67 very excited very rich
```

1 point: left `67`

1 point: middle `very excited`

1 point: right `very rich`

Hash:

4) (3 points) What is the output?

```
#!/usr/local/bin/perl5 -w
$bill{"Gates"} = "rich";
@bill{"Clinton","Mr."} = qw(very nervous);
@bill{4..7} = qw(very famous + nervous);
delete($bill{5});
delete($bill{6});
@a = sort values %bill;
%bill = reverse %bill;
print "@a\n";
```

Answer:

```
nervous nervous rich very very
```

2 points: correct items (-1 for each wrong item, up to max -2)

1 point: correct order

Perl I/O:

5) (5 points) Write a Perl program like `cat`, but that reverses the order of all the lines in the file or files specified on the command line.

For example, `cat f1 f2`, where `f1` and `f2` have the following contents:

f1	f2
1	3
2	4

would print the following:

```
4
3
2
1
```

Answer:

```
   #!/usr/local/bin/perl5 -w
2  while(<>){
1       push(@lines, $_);
   #or: @lines = (@lines, $_);
   }
2  print reverse @lines;
```

or:

```
   #!/usr/local/bin/perl5 -w
2  while(<>){
2       unshift(@lines, $_);
   #or: @lines = ($_, @lines);
   }
1  print @lines;
```

or:

```
   #!/usr/local/bin/perl5 -w
5  print reverse <>;
```

Regular Expressions:
6) (4 points) Construct a regular expression that matches any number of backslashes (\) followed by any number of astricks (*).

Answer:
`/\\*\**/`
also okay: `\\*\**`
1 point: left `\\`
1 point: left `*`
1 point: middle`\*`
1 point: final `*`

Regular Expressions:
7) (4 points) What regular expression would match a line that contains "Bill" and "Gates" in the same line (order does not matter; there can be other characters between them)?

Answer:
`/(Bill.*Gates)|(Gates.*Bill)/`
1 point: `.* between Bill and Gates`
1 point: `Bill.*Gates`
1 point: `Gates.*Bill`
1 point: `|`

give 3 out of 4 points for the following (not really a single reg. expr.): `/Bill/ && /Gates/`

Regular Expressions:
8) (4 points) What regular expression would match up to 7 a's followed by 6 of any character, followed by 5 or more +'s?

Answer:
```
     /a{0,7}.{6}(\+){5,}/
```
1 point: `a{0,7}`
1 point: `.{6}`
1 point: `\+` (parentheses optional)
1 point: `{5,}`

Regular Expressions:

9) (3 points) Construct a regular expression that matches three consecutive copies of whatever is contained in `$bill`.

Answer:

```
     /$bill$bill$bill/
```

or:

```
     /($bill){3}/
```

Regular Expressions:

10) (3 points) Construct a regular expression that matches all lines that contain all the lowercase letters (`b, i, l, l`), where the letters are in order but there may be other characters in between.

For example, the following would match:

```
     abill
     abaialala
     xxxbxxxixxxlxxxlxxx
     bbiilll
```

The following would not match:

```
     llib
     bil
     bbblll
```

Answer:

```
     /b.*i.*l.*l/
```

or:

```
     /.*b.*i.*l.*l.*/
```

2 marks if very close

1 marks if something wrong

0 marks if really wrong

Regular Expressions:

11) (4 points) What regular expression would match a line that contains "`Bill`", where "`Bill`" must be bordered by a line-end/whitespace (space, tab, newline).

Examples:

    `Bill`
`Bill Gates`
`Gates Bill`
`A Bill Gates`
`ABillGates Bill Clinton` (only second Bill matches)

Not Examples:

`aBillBilla`
`BillGates` (not example, because needs space after Bill)
`ABillGatesBill Clinton`

Answer:

```
/(^|\s)Bill(\s|$)/
```

1 point: `^|\s` (\s, not just space character)

1 point: `(^|\s)` (used parentheses)

1 point: `Bill`

1 point: `(\s|$)`

Substitution:

12) (3 points) The following substitution:

```
     s/Bill.*great/Mr. Bill is funny/
```

will produce what on the following line?

```
Bill is great. Bill Gates is greater. Mr. Bill Gates is greatest.
```

Answer:

```
Mr. Bill is funnyest.
```

1 point: `Mr. Bill`

2 points: `is funnyest.`

Functions:

13) (5 points) What is the output when the following Perl program is executed?

```
#!/usr/local/bin/perl5 -w
$n1 = 5;
$n2 = 2;
$n3 = 9;
$n1 = test($n2,$n3);
print "$n1$n2$n3";
sub test{
     my($n1,$n2) = @_;
     print "$n1$n2";
     return $n2 - $n1;
}
print "\n";
```

Answer:

29729

More control flow:

14) (3 points) What is the value of `$n` after the following Perl statements are executed?

```
$n = 3;
while($n < 10){
     $n++;
     do{
          $n += 2;
          unless($n <= 6){
               last;
          }
     }while($n < 8);
     $n += 3;
}
```

Answer: 8

More control flow:

15) (3 points) What is the value of `$n` after the following Perl statements are executed?

```
     $n = 2;
     while($n--){
          if($n > 10){ last; }
          $n += 2;
          if($n <= 3){ next;}
          $n += 3;
          if($n <= 7){ redo;}
          $n += 1;
     }
```

Answer: 12

File I/O:

16) (6 points) Write a Perl program that copies one file to another. Assume the user has included the two filenames on the command line:

```
     $ perlcopy file1 file2
     $ perlcopy letter1 backup
```

Hints: The first command-line argument can be accessed in `$ARGV[0]`, and the second in `$ARGV[1]`.

Answer:

```
     #!/usr/local/bin/perl5
     system("cp $ARGV[0] $ARGV[1]");
```

or:

```
     #!/usr/local/bin/perl5
1    open(IN, $ARGV[0]);
1    open(OUT, ">$ARGV[1]");
1    @data = <IN>;
1    print OUT @data;
1    close(IN);
1    close(OUT);
```

or:

```
     #!/usr/local/bin/perl5
1    open(IN, $ARGV[0]);
1    open(OUT, ">$ARGV[1]");
1    while(<IN>){
1         print OUT $_;
     }
1    close(IN);
1    close(OUT);
```

Directory access:

17) (13 points) Write a Perl program that recursively searches the directory specified by the user. Print out the full pathnames of any files whose filename includes the string "`bill`". For example, the pathnames to `bill2.cgi` and `testbill.html` should be printed out. After printing out the filename, remove the file (it is okay to assume you have permission to delete the file).

Answer:

```
     #!/usr/local/bin/perl5
1    $dir = $ARGV[0];
     checkdir();

1    sub checkdir(){
1         chdir("$dir");
1         foreach $f (<*>){
1              if(-f $f){
                    $_ = $f;
1                   if(/bill/){
1                        print "$dir/$f\n";
1                        unlink($f);
                    }
               }
1              elsif(-d $f && -r $f && -x $f){
   # also ok:  elsif(-d $f){
1                   my $olddir = $dir;  # save old dir
1                   $dir .= "/$f";
1                   checkdir();         # recursive call
1                   $dir = $olddir;     # restore old dir
                    chdir("$dir");
               }
          }
     }
```

HTML

18) (7 points) Write a HTML document with the following characteristics:

(1) the title "`Program One`",
(2) a heading of "`Chapter 1`", and
(3) simple text of "`Bill Gates on skates`".

Answer:

```
1    <HTML>
1    <HEAD>
1    <TITLE>Program One</TITLE>
     </HEAD>
1    <BODY>
1    <H1>Chapter 1</H1>
1    Bill Gates on Skates
or:  <P>Bill Gates on Skates</P>
1    </BODY></HTML>
```

Perl CGI Programming

19) (14 points) Write a Perl CGI program to do the following:

1. In the initial screen ask the user their name, age (use a popup menu), and allow a text area for comments.
2. Also on the initial screen, include submit and reset buttons.
3. On the result screen, show the data in the following format:

```
Your name: Andrew Horner
Your age: 34
Your comments: I like Bill!
```

Answer:

```
     #!/usr/local/bin/perl5 -w
1    use CGI qw(:standard);
1    print header();
1    print start_html("test");
1    if(param()){   # if the form has already been filled out
          my $name = param("name");
1         print p, "Your name: $name";
          my $age = param("age");
1         print p, "Your age: $age";
          my $comments = param("comments");
1         print p, "Your name: $comments";
     }else{
1         print start_form();
1         print p, "Your name: ", textfield("name");
1         print p, "Your age: ", popup_menu("age", [1..100]);
1         print p, "Comments: ", textarea("comments", "", 3, 50);
 #or:     print p, "Comments: ", textarea("comments");
1         print p, submit, reset;
1         print end_form;
     }
1    print end_html();
```

Server Push

20) (8 points) Write a Perl CGI program with Server Push that prints out all the users logged in (according to `finger`), and reloads the revised data every 5 seconds indefinitely. The output should look like the following:

Users logged in:

```
Login       Name               TTY Idle    When    Where
kato     Dr. Zoltan Kato       p3   1d Tue 14:09  cssu279.cs.ust.hk
desmond  Ng Tai Wing           p5 1:14 Thu 10:40  dmx172.resnet.ust.hk
barsky   Prof. Brian Barsky    p7  17: Tue 13:15  cssg5.cs.ust.hk
horner   Andrew Horner         p8      Mon 09:25  csnt1.cs.ust.hk
```

Answer:

```
     #!/usr/local/bin/perl5.00404 -w
1    use CGI::Push qw(:standard);

1    do_push(-next_page=>\&next_page,
          -last_page=>\&next_page,
1         -delay=>5.0);

     sub next_page {
1         my @data = ‘finger‘;
1         return start_html("who Program"),
1              h1("Users logged in:"), "\n",
1              "@data\n",
1              end_html();
     }
```