```
PERL Constants
                     Mostly like C
         12345
         -12345
         18.75
         6.02E23
         0b011011
                       (binary)
                       (octal)
         0377
         0xff
                       (hex)
         0xdead_beef
                       (more hex)
         4_294_967
         "string" (subject to variable and
                    backslash interpretation)
          'string'
                     (no interpretation)
         (1,2,3)
                          list
University of Idaho
```

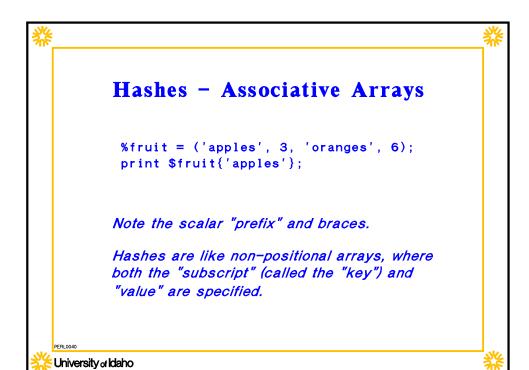
```
Scalars - a Closer Look

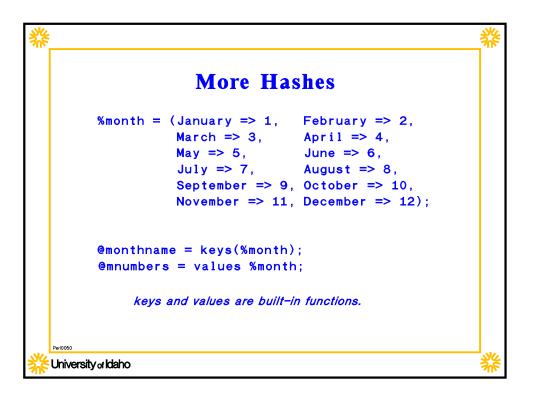
No "type" in the usual sense

$a = "123";
$b = "456";
$c = $a + $b;
print "The value of c is $c\n";
print "$a + $b is $c\n";

PERLOUSE

University of Idaho
```





```
Contexts

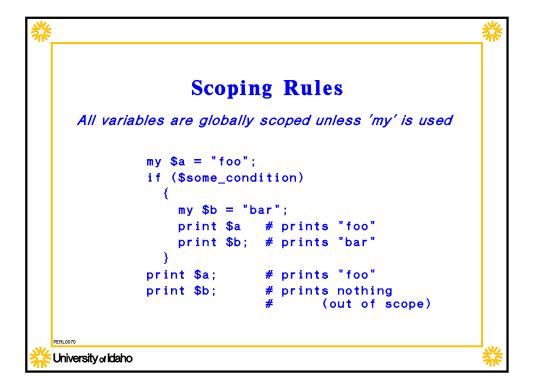
Perl "converts" values according to type of variable

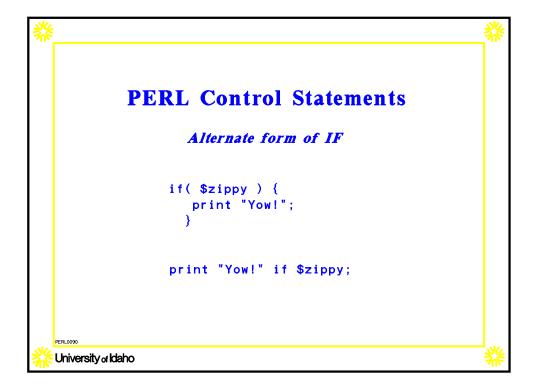
$a = (2,4,6,8);
print $a "\n"; # prints 8

@arr = (1..10);
$b = @arr; # scalar b is size of array

($sec,$min,$hour,$mday,$mon,$year,
$wday,$yday,$isdst) = localtime();

@time = localtime();
$now = localtime();
```





```
PERL Control Statements

Negative form of IF

unless (condition)
{
...
}

is equivalent to

if (!condition)
{
...
}

University oldaho
```

```
PERL Control Statements
              Conditional Comparisons
                    Op Numeric string
                   Equal
                                  eq
                Not Equal
                          !=
                                  ne
               Less than
                           <
                                  1t
             Greater than
                                  gt
             Less or equal
                        <=
                                  l e
                          >=
          Greater or equal
                                  ge
     if($str == 0 && $str ne "0") {
       warn "That doesn't look like a number\n";
University of Idaho
```

```
PERL Control Statements

while / until

while ( condition )
{
...
}

until ( condition )
{
...
}

University of Idaho
```

```
PERL Control Statements

Examples of foreach

foreach $n (1..15) {
    print $n, " ";
    }

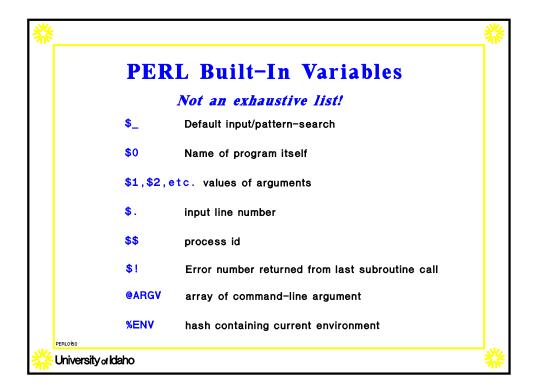
print "\n";

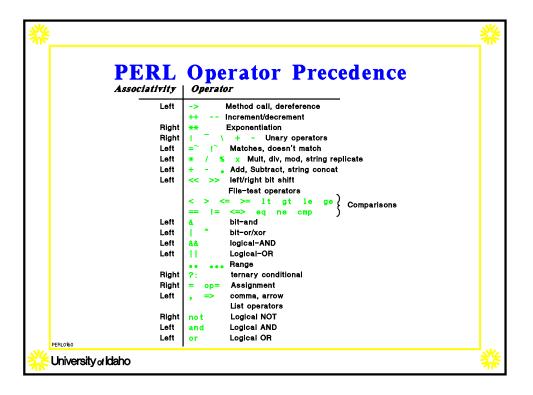
foreach my $val (keys %hashvar) {
    print "The value of $val is $hashvar{$val}\n";
    }

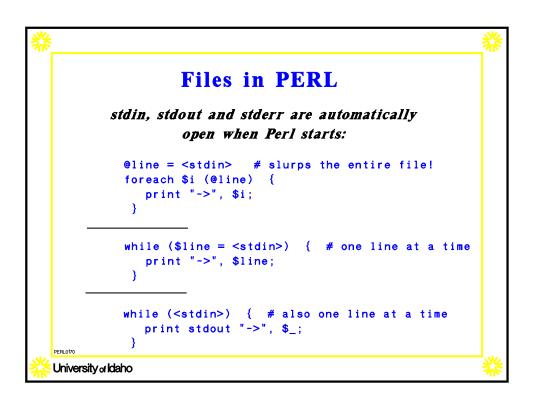
foreach (@array) {
    print "This element is $_\n";
    }

foreach (@array) {
    print;
}

whitesity of Idaho
```







```
Opening Files

$INFILE = "input.txt";
open(INFILE);

open(INFILE, "input.txt") or die "Can't open file\n";

while(<INFILE>) {
    print; # or print $_;

University aldaho
```

```
Opening Files

open(OUTFILE, ">out.txt"); #create out.txt
open(OUTFILE, ">", "out.txt");

open(LOGFILE, ">>log.txt"); # append to log.txt
open(LOGFILE, ">>", log.txt);

print OUTFILE "This goes into a file\n";

print LOGFILE "This gets appended to the file\n";

close(OUTFILE);
close(LOGFILE);

PERLORO

Whiversity of Idaho

***
```

```
PERL Strings

Concatenate

$filename = $base . ".dat";

"Contains" or "Matches"

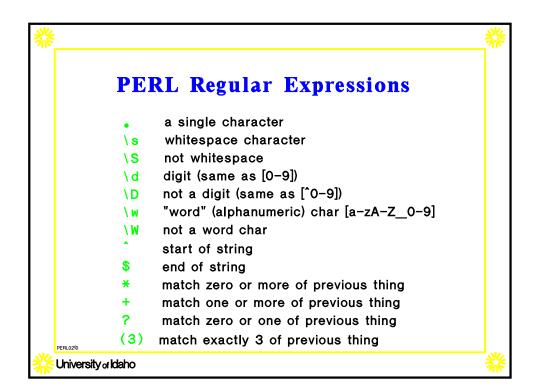
if($a =~ /foo/) { ... }

Substitution

$a =~ s/foo/bar/; # replace foo with bar
$a =~ s/foo/bar/g; # replace all

PERLOSON

Whiversity of Idaho
```



```
Regular Expression Examples
       /^\d+/
                    string starts with one or more digits
                    null string
       /(\d\s)(3) / three digits, each followed by whitespace
                    every other char is letter a
       /(a.)+/
       warn "has nondigits" if /\D/;
       warn "not an integer" unless /^[+-]?\d+$/
       while (<>) {
         next if /^$/;
          print;
 LINE: while(<STDIN>) {
          next LINE if /^#/;
           #do something with line
University of Idaho
```

```
PERL Subroutines

sub max {
    my $maxval = @_[0];
    foreach $val (@_) {
        $maxval = $val if $val > $maxval;
      }
    return $maxval;
}

• Subroutines are defined with 'sub' keyword
• No formal arguments: arguments passed in array @__
• Two ways to call:
    max($a, $b, $c);
    max $a $b $c; #only if defined before call!

***
University aldaho
```

```
PERL "Built—In" Functions

File test Functions - return "true" if file:

-r file is readable
-w file is writeable
-x file is executable
-o file is owned by user
-e file exists
-z file is zero bytes in length
-f file is a regular file
-d file is a directory
-1 file is a symlink
```



PERL "Built-In" Functions - Part 2

Exit Functions

exit val return to shell, returning val
die message print message and exit, returning \$!
warn message just print message

Array Functions

shift @arr return first element, shift others down

Hash Functions

keys %hash return array of all key values

values %hash return array of all values of keys

University of Idaho





PERL "Built-In" Functions - Part 3

String Functions

chomp string

Remove input separator (\$/, usually newline) from string

index @str, substr

Return position of substr within array str

join char, list

Return single string consisting of list separated by char

length string

Return no of characters in string

q/string/ q//string/

Alternate form of 'string' and "string"

split /pattern/, string

Return list of substrings within string separated by pattern

PERL0270

University of Idaho

