

Sorting Letters and Numbers

by Dr. David J. Ritchie,
Computer Club Advisor

The perl function called sort will let you arrange things in alphabetical or numerical order.

Try sorting numbers:

1. Write this program:

```
@b = (5, 4, 3);  
@a = sort @b;  
print "@a";
```

2. Run it.

3. Did it sort the array?

Try sorting letters:

1. Write:

```
@b = (C, X, A);  
@a = sort @b;  
print "@a";
```

2. Run it.

3. Are your numbers sorted?

INFO

- Arrays store many things.
- Arrays begin with @.
- Things are called elements.
- Each element is given a number from 0 onwards.
- Elements begin with \$ and have their number in [].
- Example: @A, \$A[3].

An easy way to put something into the backpack is to use push. Example:

```
push @Pack, "gold chain";
```

This will push the literal "gold chain" into the array @Pack. To remove the last thing pushed, use pop. Example:

```
pop @Pack;
```

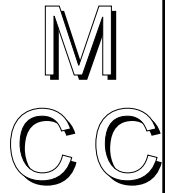
Issue 3

January 10, 2000

Madison Computer Club News

Mrs. Gilmore

Dr. David J. Ritchie



Students Extend Adventure Game!

Students to add knapsack next!

Members of the Madison Computer Club successfully extended their Adventure Games in a variety of ways.

By adding a line to print status in the basic game loop (see the status line in the insert titled "The Basic Adventure Game Loop" on the reverse side), the program can tell the player the status of the game. (In the example, the status line shows the page number on which the player is located.)

Let's make a backpack for a player to carry in the game. Since a backpack carries more than one thing, we will use an array. We might as well call it @Pack.

More about backpacks...

Now that we have a backpack, let's work it into our story by having a page to pick up something, to lose it all, and so on.

Now, let's do a page where we lose everything in the backpack. Example:

The Basic Adventure Game Loop..

```
#
# Do Adventure Game Loop...
while ($Page > 0 ) {
    print "===== Page: $Page\n";      #status
    if ($Page == 1) { $NextPage = Page1();}
    if ($Page == 2) { $NextPage = Page2();}
    if ($Page == 3) { $NextPage = Page3();}
    if ($Page == 4) { $NextPage = Page4();}
    if ($Page == 5) { $NextPage = Page5();}
    if ($Page == 6) { $NextPage = Page6();}
    if ($Page == 7) { $NextPage = Page7();}
    if ($Page == 8) { $NextPage = Page8();}
    if ($Page == 9) { $NextPage = Page9();}
    if ($Page == 10) { $NextPage = Page10();}
    $Page = $NextPage
}
#
```

```
sub Page12 {
    print <<END;
    You have fallen into a
    pit, lost all your things,
    and have to start over.
    END
    @Pack = ();
    $Choice = 1;
    return ($Choice);
}
```

What about just losing the gold? We have to look through our pack, find it, and throw it away.

Example:

First, let's print the contents of the backpack on the status line. But we'd better print it in a sorted way. Example:

```
@SortedPack = sort @Pack;
print "=====Page: $Page\n",
      "Pack: @SortedPack\n";
```

Now, let's have a page to pick up a thing for our backpack. Example:

```
sub Page11 {
    print <<END;
    You've come across some gold.
    END
    $Choice = Choose("Pick it up", 2,
                    "Leave it alone", 3);

    if ($Choice == 2) {push @Pack, "gold";}
    return ($Choice);
}
```

```
sub Page12 {
    print <<END;
    You have lost the gold. Back to Page 3.
    END
    for ($i = 0; $i <= $#Pack; $i++) {
        if ($Pack[$i] eq "gold") {
            $Pack[$i] = " ";
        }
    }
    $Choice = 3;
    return ($Choice);
}
```

Or, use the splice subroutine. Example:

```
sub Page12 {
    print <<END;
    You have lost the gold. Back to Page 3.
    END
    for ($i = 0; $i <= $#Pack; $i++) {
        if ($Pack[$i] eq "gold") { $here = $i; }
    }
    splice @Pack, $here, 1;
    $Choice = 3;
    return ($Choice);
}
```