

Beginning Perl Programming (X52.9543.001)**An Implementation of Homework 2**

Here's **one** implementation of Homework #2. The **bold** items highlight changes/improvements from savbond1.pl:

savbond2.pl

```
#!/App/Perl/bin/perl -w
#
# savbond2.pl
#
# Perl script to show the current value of a US Savings Bond or Note.
# It works interactively on STDIN/STDOUT and works for Series E Bonds,
# Series EE Bonds, and Savings Notes. Requires a data file downloadable from
# http://www.publicdebt.treas.gov/sav/savvalue.htm.
# These files are typically of the form SByyyyymm.ASC, where yyyymm
# represents a recent year-month.
#
# (c) Copyright 2000 by Clint Goss, All Rights Reserved.

print "Welcome to Clint's Savings Bond Calculator (version 2)!\n\n";
print "This program calculates the current value of your\n";
print "Series E or EE US Savings Bond, or your US Savings Note.\n\n";

# Ask for the bond type and convert to our standard types

print "Please enter 'E' or 'EE' for a Series E or EE US Savings Bond,\n";
print "or 'S' for a US Savings Note: ";

chomp ($userBondType = <STDIN>);
print "\n";

$userBondType = uc ($userBondType);

$bondType = $userBondType;
if ($bondType eq "EE") { $bondType = "N"; }

# TBD: Check that bond type is one we know about

# Ask for the bond face value

print "Please enter the denomination or face value of your Bond or Note\n";
print "(eg. for a \$50 Bond, enter 50): ";
chomp ($bondDenomination = <STDIN>);
print "\n";

# TBD: Input checking/munging - remove leading '$'??

# Ask for the month of issue and parse out the response

print "Please enter the Month/Year your bond was issued (eg. 05/1984): ";
chomp ($bondIssueDate = <STDIN>);
print "\n";

($bondIssueMonth, $bondIssueYear) = split ("/", $bondIssueDate);

# TBD: Input range checking, conversion of 2-digit years to 4-digit??.
X52.9543_Summer2000_Homework02.doc
```

```

# TBD: Accept human-type input like "Jan" or "January".

# Determine the current month and year

@timeComponents = localtime(time);
$currentMonth = $timeComponents[4] + 1;
$currentYear = $timeComponents[5] + 1900;

# Open our data file
# KLUGE: Hard-coded file name

$dataFileName = "SB200003.ASC";
open (DATAFILE, "<$dataFileName") || die "Can't open $dataFileName";

# Go looking for the correct line in the file

# Column position:                1111111111222 ...
# Column position:                01234567890123456789012 ...
# Sample line (truncated): E2000031971011868011896 ...
# Field codes:                    tyyyyymmiiiijjjjjjfffff ...
#
# Explanation of field codes:
#   t      bond type (E=Series E Savings Bond,
#           N=Series EE Savings Bond, S=Savings Note)
#   yyyy  redemption year
#   mm    redemption month
#   iiii  year of issue
#   jjjjj Value in cents of a $25 Bond/Note issued in January
#   fffff Value in cents of a $25 Bond/Note issued in February

$foundAtLeastOneRedemptionMonthYear = 0;

while ($record = <DATAFILE>) {

    # Carve up this line of data into it's fields.

    chomp ($record);
    $recBondType = substr ($record, 0, 1);
    $recRedemptionYear = substr ($record, 1, 4);
    $recRedemptionMonth = substr ($record, 5, 2);
    $recIssueYear = substr ($record, 7, 4);

    # Skip this line if the redemption year/month does not match the
    # current year/month, or if we've got the wrong bond type.

    if ($recRedemptionYear != $currentYear) { next; }
    if ($recRedemptionMonth != $currentMonth) { next; }

    # Remember that this file has at least one record which covers the
# current month/year.

    $foundAtLeastOneRedemptionMonthYear = 1;

    if ($recBondType ne $bondType) { next; }

    # Skip this line if it does not match the correct year of issue

    if ($recIssueYear != $bondIssueYear) { next; }

    # Here only if we have the correct line of data

```

```

# Pull out the correct 6 characters, based on the month of issue.

$valueStartPosition = 11 + (6 * ($bondIssueMonth - 1));
$valuePicture = substr ($record, $valueStartPosition, 6);

if ($valuePicture eq (" " x 6)) {

    printf "No bond has yet been issued for month/year %02d/%d.\n",
        $bondIssueMonth, $bondIssueYear;

} elsif ($valuePicture eq "NO PAY") {

    printf "Bonds issued in month/year %02d/%4d are " .
        "not yet eligible for payment.\n",
        $bondIssueMonth, $bondIssueYear;
} else {

    # Compute the value - convert picture from cents to dollars and
    # scale answer to correct bond denomination

    $value = $valuePicture / 100 * ($bondDenomination / 25);

    # Display the output and exit!

    printf "Your Bond / Note is currently worth \\\$%0.2f.\n", $value;
}

exit;
}

# Here only if none of the lines in the data file matched

print "Problem: ";

if (! $foundAtLeastOneRedemptionMonthYear) {
    printf "The file %s does not cover the current month/year %02d/%d.\n",
        $dataFileName, $currentMonth, $currentYear;
} else {
    if ($bondType eq "S") {
        printf "No US Savings Notes";
        printf " were issued in the month/year %02d/%d.\n",
            $bondIssueMonth, $bondIssueYear;
    } else {
        printf "No Series %s US Savings Bonds", $userBondType;
        printf " were issued in the month/year %02d/%d.\n",
            $bondIssueMonth, $bondIssueYear;
    }
}
}

```