



**(National Council for Vocational Awards)**



# **Computer Programming C20013**

**Sample Theory Examination 2007**

## **Duration: Two Hours**

**INSTRUCTIONS TO CANDIDATES:**

*Answer any **three** questions*

*All questions carry equal marks*

**Answer the questions using the spaces in this exam booklet**

**Return this question & answer paper when finished**

**This sample written exam counts as 40% of the total module**

NAME (PRINT): SAMPLE

PPS NUMBER: \_\_\_\_\_

DATE: \_\_\_\_\_

**Question 1. Total 40 marks.**

(a) This program contains 4 errors that will stop it from compiling. List the errors.

**20 marks**

```
#!/usr/bin/perl
# A greeting program!
Print "Hello, there. What is you name? ";
$punters_name=<STDIN>;
print <nHello, $punters_name.\nHave a nice day!\n>;
```

1	Miss-spelling of perl
2	Should be small 'p' for perl
3	extra > after <STDIN>
4	Unmatch inverted commas - none at start of print.

(b) In all languages variables have to be declared before use. True or False? **10 marks**

FALSE. Variables need not be declared before use in perl, BASIC and some other languages.

(c) Does a Perl array variable commence with a % or a \$ symbol? **10 marks**

@ symbol

**Question 2. Total 40 marks.**

(b) Write the general form of the method to read lines of data from a file **10 marks**

```
#file = "somefile.dat"; # declare a file name
open (AHANDLE, "< $file"); # connect a "file handle" to
@Array = <AHANDLE>; # Put everything from the file in an
close (AHANDLE); # Close the file by array
                     closing the handle.
```

(b) Write the general form of the **foreach** statement: **10 marks**

```
foreach $oneRecord (@ lots of records)
{
    Do stuff with $one Record;
}
```

(c) The following perl code will compile and run but will not generate the desired output. Why?

**2 x 10 marks**

```
#!/usr/bin/perl
$filename="myrecords.dat";
open (MyDataFile, "< $filename");
@The_Records = <MyDataFile>;
close (MyDataFile);
# This program should write out the
# contents of a file on numbered lines.
```

```
$n=1;
foreach $record (@The_Records)
```

```
{
    print "n: %record\n";
    $n++;
}
```

Should be  
\$n      Should be \$record

**Question 3. Total 40 marks.**

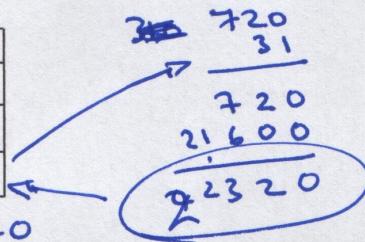
- (a) Indicate the values in each of the variables \$x, \$y and \$z after this program finishes:

**30 marks**

```
#!/usr/bin/perl
$num = 0;
$x = 310;
while ($num <= 5) → loop ends at 6
{
    $num=$num+1;
    $x = $x;
}
$y = $num * $num;
$z = 12 * 6 * $x;

print ("$x, $y, $z\n");
```

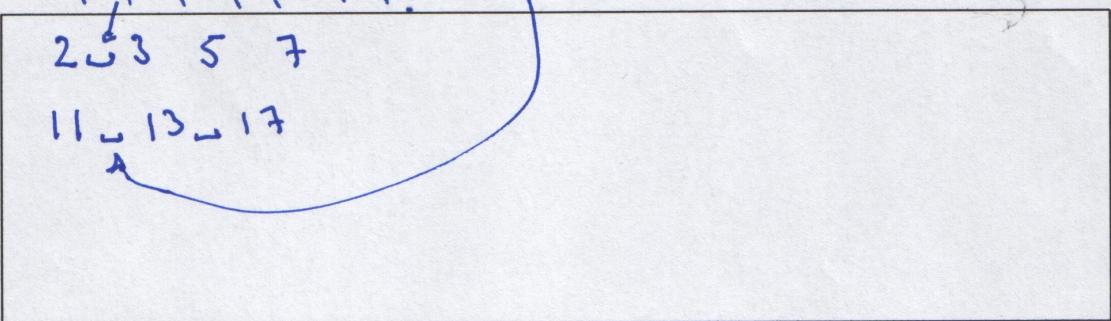
Variable	Value
\$x	310
\$y	36
\$z	$72 \times 310$



- (b) What screen output is generated by this short program:

**10 marks**

```
#!/usr/bin/perl
printf ("%c%c%c%c%c\n%c%c%c%c%c\n",
50,32,51,32,53,32,55,49,49,32,49,51,32,49,55);
```



**Question 4. Total 40 marks.**

- (a) Write a perl loop to write out every odd number between 11 and 101 and then write out the total of the numbers which have been printed.

**30 marks**

```
$counter = 11;  
$total = 0;  
while ($counter <= 101)  
{  
    print "$counter\n";  
    $counter = $counter + 2; # skips the even No.s.  
    $total = $total + $counter;  
}  
print "The total is: $total\n";
```

- (b) If you are to use a **while** loop or a **for** loop; what generally is considered to be the difference between both loop types?

**10 marks**

With a **for** loop you generally know in advance how many times the loop will iterate. With a **WHILE** Loop the loop could go for a short or a long run - you don't know at first.