PERL UNPACK FUNCTION

Description

This function unpacks the binary string STRING using the format specified in TEMPLATE. Basically reverses the operation of pack, returning the list of packed values according to the supplied format.

You can also prefix any format field with a %<number> to indicate that you want a 16-bit checksum of the value of STRING, instead of the value.

Syntax

Following is the simple syntax for this function −

```
unpack TEMPLATE, STRING
```

Return Value

This function returns the list of unpacked values.

Here is the table which gives values to be used in TEMPLATE.

Character	Description
a	ASCII character string padded with null characters
A	ASCII character string padded with spaces
b	String of bits, lowest first
B	String of bits, highest first
c	A signed character $range usually - 128 to 127$
C	An unsigned character $usually 8 bits$
d	A double-precision floating-point number
f	A single-precision floating-point number
h	Hexadecimal string, lowest digit first
H	Hexadecimal string, highest digit first
i	A signed integer
I	An unsigned integer
l	A signed long integer
L	An unsigned long integer
n	A short integer in network order
N	A long integer in network order
p	A pointer to a string
s	A signed short integer
S	An unsigned short integer

u	Convert to uuencode format
v	A short integer in VAX $little - endian$ order
V	A long integer in VAX order
x	A null byte
X	Indicates "go back one byte"
@	Fill with nulls $ASCII0$

Example

Following is the example code showing its basic usage −

```
#!/usr/bin/perl -w

$bits = pack("c", 65);
# prints A, which is ASCII 65.
print "bits are $bits\n";
$bits = pack( "x" );
# $bits is now a null chracter.
print "bits are $bits\n";
$bits = pack( "sai", 255, "T", 30 );
# creates a seven charcter string on most computers'
print "bits are $bits\n";

@array = unpack( "sai", "$bits" );

#Array now contains three elements: 255, A and 47.
print "Array $array[0]\n";
print "Array $array[1]\n";
print "Array $array[2]\n";
```

When above code is executed, it produces the following result −

```
bits are A
bits are
bits are �T
Array 255
Array T
Array 30
```