Code No: 156CU **R18**

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

B. Tech III Year II Semester Examinations, February - 2023

SCRIPTING LANGUAGES

(Common to CSE, IT)

Time: 3 Hours **Max. Marks: 75**

Note: i) Question paper consists of Part A, Part B.
ii) Part A is compulsory, which carries 25 marks. In Part A, Answer all questions.
iii) In Part B, Answer any one question from each unit. Each question carries 10 marks and may have a, b as sub questions.

PART – A

(25 Marks)

1.

a) What do you mean by an iterator in Ruby? [2]

b) State the importance of RubyGems. [3]

c) Write about Fixnum immediate value in Ruby. [2]

d) Write an extension code in C that is plug-compatible with that Ruby class. [3]

e) Define the term Web scripting. [2]

f) How does chop( ) and chomp( ) functions in PERL differ? [3]

g) Define a module and a package. [2]

h) Give a brief file system operations in PERL. [3]

i) How to create a list in TCL? Give an example. [2]

j) What is the purpose of ‘exec’ and ‘append’ commands in Tcl? [3]

PART – B

(50 Marks)

2.a) “Embedding Ruby in HTML is a very powerful concept”. Support this statement using code snippets.

b) How to access Tk features from Ruby? Discuss briefly creating a simple application. [5+5]

OR

3. How Ruby CGI handles cookies and sessions? Demonstrate using the sample Ruby CGI Scripts. [10]

4. Explain various C/Ruby data type conversion functions and macros with clear syntax and illustrative examples for each. [10]

OR

5.a) Explain StringValue and StringValuePtr macros in Ruby with sample Ruby script.
b) How to embed ruby in your application? What is the role of Ruby interpreter? Explain. [4+6]

6. Explain various regular expressions and pattern matching operators PERL with examples. available in [10]

OR

7.a) What are the uses of scripting languages?
b) Explain various string and numeric built-in functions available in PERL. [4+6]

8.a) Outline the security issues in internet programming of PERL.
b) Give a detailed note on using PERL interfacing to an operating system. [5+5]

OR

9.a) How to create packages and modules in PERL? Describe in detail.
b) What is the need of pack( ) and unpack( ) functions in Perl. [6+4]

10. What are the major functionalities of a Tcl Parser? Explain in detail about namespaces and recursive procedures in TCL. [10]

OR

11.a) Write Tk script to create a login form.
b) How are events handled in Tk? [5+5]

---ooOoo—

Answer Key

PART - A

1.a) What do you mean by an iterator in Ruby?

An iterator in Ruby is a method that yields successive elements from a collection to a block of code. It allows traversal and manipulation of elements in data structures like arrays and hashes. Common iterators include `each`, `map`, and `select`.

1.b) State the importance of RubyGems.

RubyGems is a package management framework for Ruby that simplifies the distribution, installation, and version management of Ruby libraries. It facilitates dependency management and allows developers to share and reuse code efficiently. It enhances productivity and code maintainability in Ruby projects.

1.c) Write about Fixnum immediate value in Ruby.

Fixnum represents integer values that are embedded directly into the Ruby interpreter's internal representation, making them highly efficient. Immediate values like Fixnum do not require object allocation, leading to faster arithmetic operations. In Ruby 2.4 and later, Fixnum is unified with Bignum into the Integer class.

1.d) Write an extension code in C that is plug-compatible with that Ruby class.

```c
#include "ruby.h"
VALUE MyClass;
VALUE my_method(VALUE self) {
   return rb_str_new2("Hello from C");
}

void Init_my_extension() {
   MyClass = rb_define_class("MyClass", rb_cObject);
   rb_define_method(MyClass, "my_method", my_method, 0);
}
```

1.e) Define the term Web scripting.

Web scripting refers to writing scripts, usually in languages like JavaScript, PHP, or Ruby, to create dynamic web content. It enables interactive functionalities on web pages by running code on either the client-side or server-side. Web scripting enhances user experience by allowing real-time updates and data manipulation.

1.f) How does chop() and chomp() functions in PERL differ?

The `chop()` function removes the last character of a string, regardless of its value. The `chomp()` function removes the newline character (`\n`) at the end of a string if it exists. `chomp()` is safer for removing line endings, while `chop()` is more general.

1.g) Define a module and a package.

A module is a reusable, encapsulated piece of code that can include methods, constants, and classes, used to organize and share functionality. A package is a namespace that organizes related modules and classes, preventing name conflicts and facilitating code distribution.

1.h) Give a brief file system operations in PERL.

Perl provides functions like `open`, `close`, `read`, `write`, and `unlink` for file handling. It supports file operations such as creating, reading, writing, and deleting files. Filehandles are used to interact with files, enabling efficient input and output operations.

1.i) How to create a list in TCL? Give an example.

In TCL, lists can be created using the `list` command. Example:

```tcl
set mylist [list 1 2 3 4 5]
```

This creates a list with elements 1, 2, 3, 4, and 5.

j) What is the purpose of ‘exec’ and ‘append’ commands in Tcl?

The `exec` command runs an external program or script and captures its output. The `append` command adds a string to the end of an existing variable's value. Both commands are used to manage external processes and string manipulation efficiently.

PART - B

2.a) “Embedding Ruby in HTML is a very powerful concept”. Support this statement using code snippets.

Embedding Ruby in HTML allows for dynamic content generation, making web pages more interactive and responsive to user inputs. This concept is commonly used in Ruby on Rails, a popular web application framework.

#Example using Embedded Ruby (ERB):

~~~
```html
<!DOCTYPE html>
<html>
<head>
  <title>My Ruby Page</title>
</head>
<body>
  <h1>Welcome to my Ruby Page!</h1>
  <% if @user %>
    <p>Hello, <%= @user.name %>!</p>
  <% else %>
    <p>Hello, Guest!</p>
  <% end %>
</body>
</html>
``
~~~

In this example:

- `<% %>` is used for Ruby code that doesn't produce output.
- `<%= %>` is used for Ruby code that produces output.

2.b) How to access Tk features from Ruby? Discuss briefly creating a simple application.

Tk is a GUI toolkit that can be accessed from Ruby using the `tk` gem. Here is a brief example of a simple Tk application:

1. Install the Tk gem:

~~~
```sh
gem install tk
```
~~~

2. Create a simple Tk application:

~~~
```ruby
require 'tk'
root = TkRoot.new { title "Simple Tk App" }
TkLabel.new(root) do
  text 'Hello, Tk!'
~~~

```
  pack { padx 15 ; pady 15; side 'left' }
end
Tk.mainloop
```

This script creates a window with the title "Simple Tk App" and a label that says "Hello, Tk!".

3. How Ruby CGI handles cookies and sessions? Demonstrate using the sample Ruby CGI Scripts.

Ruby CGI handles cookies and sessions to maintain state between HTTP requests. Here’s how you can manage cookies and sessions in a Ruby CGI script:

1. Handling Cookies:

```ruby
require 'cgi'
cgi = CGI.new
cookie = CGI::Cookie.new('name' => 'user', 'value' => 'Dattasai')
cgi.out('cookie' => cookie) do
  "Cookie has been set."
end
```

2. Handling Sessions:

```ruby
require 'cgi'
require 'cgi/session'
cgi = CGI.new
session = CGI::Session.new(cgi)
session['user'] = 'Dattasai'
session.close
cgi.out do
  "Session data saved."
end
```

4. Explain various C/Ruby data type conversion functions and macros with clear syntax and illustrative examples for each.

In Ruby C extensions, converting between C and Ruby data types is essential. Here are some common conversion functions and macros:

1. NUM2INT: Converts a Ruby number to a C integer.

```
```

```c
int c_value = NUM2INT(ruby_value);
```

2. INT2NUM: Converts a C integer to a Ruby number.

```c
VALUE ruby_value = INT2NUM(c_value);
```

3. StringValueCStr: Converts a Ruby string to a C string.

```c
const char* c_string = StringValueCStr(ruby_string);
```

4. rb_str_new2: Creates a Ruby string from a C string.

```c
VALUE ruby_string = rb_str_new2(c_string);
```

5.a) Explain `StringValue` and `StringValuePtr` macros in Ruby with sample Ruby script.

- StringValue: Ensures the given value is a Ruby string.

```c
VALUE str = rb_str_new2("Hello");
StringValue(str); // Now str is a Ruby string
```

- StringValuePtr: Converts a Ruby value to a C string.

```c
VALUE str = rb_str_new2("Hello");
char* c_str = StringValuePtr(str);
```

5.b) How to embed Ruby in your application? What is the role of Ruby interpreter? Explain.

To embed Ruby in an application, you can initialize and interact with the Ruby interpreter from your C code. The Ruby interpreter executes Ruby code, providing access to its libraries and runtime environment.

Example of embedding Ruby:

```c
#include <ruby.h>
int main(int argc, char argv) {
  ruby_init();
  ruby_init_loadpath();
  rb_eval_string("puts 'Hello from embedded Ruby!'");
  ruby_cleanup(0);
  return 0;
}
```

6. Explain various regular expressions and pattern matching operators in PERL with examples.

Perl is renowned for its powerful regular expressions and pattern matching capabilities.

1. Matching operator (`=~`):

```perl
my $string = "Hello World";
if ($string =~ /World/) {
  print "Match found!\n";
}
```

2. Substitution operator (`s///`):

```perl
$string =~ s/World/Perl/;
print "$string\n"; # Output: Hello Perl
```

3. Translation operator (`tr///`):

```perl
$string =~ tr/aeiou/AEIOU/;
print "$string\n"; # Output: HEllO WOrld
```

7.a) What are the uses of scripting languages?

Scripting languages are used for:

- Automating repetitive tasks
- Rapid application development
- Web development (server-side and client-side)
- System administration
- Text processing and data analysis

7.b) Explain various string and numeric built-in functions available in PERL.

String functions:

- `length`: Returns the length of a string.
- `substr`: Extracts a substring.
- `index`: Finds the position of a substring.

```perl
my $str = "Hello World";
print length($str); # 11
print substr($str, 6, 5); # World
print index($str, "World"); # 6
```

Numeric functions:

- `abs`: Returns the absolute value.
- `int`: Converts a value to an integer.
- `rand`: Generates a random number.

```perl
print abs(-10); # 10
print int(3.14); # 3
print rand(10); # Random number between 0 and 10
```

8.a) Outline the security issues in internet programming of PERL.

Security issues in Perl internet programming include:

- Injection attacks: Through user inputs.
- File access vulnerabilities: Unauthorized access to files.
- Unsafe eval usage: Running arbitrary code.

8.b) Give a detailed note on using PERL interfacing to an operating system.

Perl can interface with the OS using built-in functions and external modules:

- System commands: `system`, `` (backticks)
- File handling: `open`, `read`, `write`
- Modules: `POSIX`, `File::Find`

Example:

```perl
# Execute a system command
system("ls -l");
# Read a file
open(my $fh, '<', 'file.txt');
while (<$fh>) {
  print $_;
}
close($fh);
```

9.a) How to create packages and modules in PERL? Describe in detail.

Creating a package and module in Perl:

1. Package:
```perl
package MyPackage;
sub hello {
  print "Hello from MyPackage!\n";
}
1; # Return true to indicate the package loaded successfully
```
2. Module (save as `MyModule.pm`):
```perl
package MyModule;
use strict;
use warnings;
sub hello {
  print "Hello from MyModule!\n";
}
1;
```
Use the module:
```perl
use MyModule;
MyModule::hello();
```

9.b) What is the need of `pack` and `unpack` functions in Perl.

`pack` and `unpack` are used for binary data manipulation.

- pack: Converts a list into a binary representation.

```perl
my $binary_data = pack("C*", 65, 66, 67);
```

- unpack: Converts binary data back into a list.

```perl
my @array = unpack("C*", $binary_data);
```

10. What are the major functionalities of a Tcl Parser? Explain in detail about namespaces and recursive procedures in TCL.

Tcl Parser functionalities:

- Tokenizes Tcl scripts
- Handles syntax and semantic checks

- Executes Tcl commands

Namespaces:

Namespaces in Tcl provide a way to group related procedures and variables, avoiding name conflicts.

```tcl
namespace eval myNamespace {
  variable myVar 10
  proc myProc {} {
    variable myVar
    puts $myVar
  }
}
myNamespace::myProc
```

Recursive procedures:

```tcl
proc factorial {n} {
  if {$n <= 1} {
    return 1
  } else {
    return [expr {$n * [factorial [expr {$n - 1}]]}]
  }
}
puts [factorial 5] # Output: 120
```

11.a) Write Tk script to create a login form.

```tcl
package require Tk
set top .top
toplevel $top
wm title $top "Login Form"
label $top.lbl1 -text "Username:"
entry $top.ent1
label $top.lbl2 -text "Password:"
entry $top.ent2 -show "*"
button $top.btn -text "Login" -command {
  puts "Username: [$top.ent1 get]"
  puts "Password: [$top.ent2 get]"
}
```

```
grid $top.lbl1 $top.ent1
grid $top.lbl2 $top.ent2
```