

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 1: Overview of COMP284
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

Assignment Project Exam Help

① Introduction

Motivation

Scr

② COM

Aims

Learning outcomes

Delivery

Assessment

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

How many programming languages should you learn?

- ① Academic / Educational viewpoint:

Learn programming language concepts and

use programming languages to gain practical experience with them

- imperative / object-oriented
 - C, Java

- functional
 - Maude, OCaml

- logic
 - P

- coroutines

- then a

- ② An employer's viewpoint:

Learn exactly those programming languages that

needs

- ③ Compromise: Spend most time on ① but leave some time for ② to

allow more than one language from a class/paradigm to be learned

- ④ Problem: Which additional language do you cover?

→ Look what is used/demanded by employers

Programming languages: Job ads

Software Developer
(Digital Repository)

University of Liverpool - University Library

£31,020 - £35,939 pa



Assignment Project Exam Help

To work as p

with the b
repositor

<https://eduassistpro.github.io>

functionality to integrate the repository with other intern

research outputs to be shared externally. You will be an exper

Developer with knowledge of TAMP technologies such as

Javascript. You will hold a degree in Computer Science or a rel

and/or have proven industrial experience of software development. The post is full time, 35 hours per week.

Job Ref: A-576989

Programming languages: Job ads

Senior Software Development Manager

IMDb Video and Recommendations (Seattle, WA)

MPL (a wholly-owned subsidiary of Amazon) is recruiting for a Senior Software Development Manager to lead our "What to Watch" team. You'll be charged with transf millions of p providers best suited f

<https://eduassistpro.github.io>

Basic qualifications:

- Bachelor's degree in Computer Science, Computer related technical discipline
- 10+ years of experience as a software developer
- 5+ years experience managing people
- Software development experience in OOP, Java, Perl, HTML, CSS, JavaScript, Linux/UNIX, AJAX, MySQL

Programming languages: Job ads

Full-time Remote Worker

AOL Tech (Engadget, TUAW, Joystiq, Massively)

AOL Tech is looking for a great front-end developer who can help us take Engadget and our other blogs to new levels.

The ideal candidate is highly proficient in **JavaScript/jQuery**, comfortable with **PHP / MySQL**

technologies
is a must.

<https://eduassistpro.github.io/>

Requirements:

- High proficiency in **JavaScript/Query**
- Familiar with spriteing, lazy loading, and other general performance-optimized techniques
- Mac access for compatibility with current tools
- HTML5/CSS3
- Git, SSH

Add WeChat **edu_assist_pro**

Websites and Programming Languages

Website	Client-Side	Server-Side	Database
Google	JavaScript	C, C++, Go, Java, Python, PHP	BigTable, MariaDB
Facebook			, MySQL, sandra
YouTube			, MariaDB
Yahoo	JavaScript	PHP	reSQL
Amazon	JavaScript	Java, C++, Perl	Oracle DB
Wikipedia	JavaScript	PHP, Hack	MySQL
Twitter	JavaScript	C++, Java, Scala	MySQL
Bing	JavaScript	ASP.NET	MS SQL Server

Wikipedia Contributors: Programming languages used in most popular websites. Wikipedia, The Free Encyclopedia, 20 October 2017, at 11:28. http://en.wikipedia.org/wiki/Programming_languages_used_in_most_popular_websites [accessed 23 October 2017]

Scripting languages

Script

A user-readable and user-modifiable program that performs simple operations and controls the operation of other programs

Scriptin

A progra

<https://eduassistpro.github.io/>

Classical example: Shell scripts

```
#!/bin/sh
for file in *; do
    wc -l "$file"
done
```

Add WeChat edu_assist_pr

Print the number of lines and name for each file in the current directory

Scripting languages: Properties

- Program code is present at run time and starting point of execution
 - compilation by programmer/user is not needed
 - compilation to bytecode or other low-level representations may be performed 'behind the scenes' as an optimisation
- Presence of scripts
 - includes virtual machines
 - typically also includes a large collection of libraries
- Execution of scripts is typically slower than the executable has been fully pre-compiled to machine code

```
#!/bin/sh
for file in *; do
    wc -l "$file"
done
```

Assignment Project Exam Help
<https://eduassistpro.github.io/>
Add WeChat edu_assist_pro

Scripting languages: Properties

- Rich and easy to use interface to the underlying operating system, in order to run other programs and communicate with them
 - Rich input/output capabilities, including pipes, network sockets, file I/O, and filesystem operations

- Easy in

- often
- can do

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

```
#!/bin/sh
for file in *; do
    wc -l "$file"
done
```

Scripting languages: Properties

- Variables, functions, and methods
typically do not require type declarations

(automatic conversion between types, e.g. strings and numbers)

- Some built-in data structures

(more t

- Ability to interact with the environment through the browser's API

<https://eduassistpro.github.io/>

JavaScript

```
var x = 2;  
var y = 6;  
var str = "if (x > 0) { z = y / x } else { z = -1 }";  
console.log('z is ', eval(str)); // Output: z is 3  
x = 0;  
console.log('z is ', eval(str)); // Output: z is -1
```

Scripting languages: Properties

- The evolution of a scripting language typically starts with a limited set of language constructs for a specific purpose
Example: PHP started as set of simple 'functions' for tracking visits to a web page

- The language evolves as it is used

- These changes may conflict with the original core and/or may duplicate existing features

- During this evolution of the language, backward compatibility may or may not be preserved

→ Language design of scripting languages is often sub-optimal

Aims

Assignment Project Exam Help

① To provide students with an understanding of
the na

② To in <https://eduassistpro.github.io>
and t

③ To enable students to write simple scripts using th
for a variety of applications

Add WeChat edu_assist_pr

Learning Outcomes

At the end of the module students should be able to

Assignment Project Exam Help

- ① compare and contrast languages such as JavaScript, Perl and PHP with other programming languages

② docu

<https://eduassistpro.github.io>

③ rapi

using an appropriate scripting language

Add WeChat edu_assist_pro

Delivery of the module (1)

① Lectures

- Structure:

16 to 18 lectures

- Schedule:

1 or 2 lectures per week spread over 9 weeks

See

<https://eduassistpro.github.io/>

- Lectures

cgi.csc.liv.ac.uk/~ullrich/C

Add WeChat edu_assist_pro

- Revise the lectures before the corresponding [prac](#)
- Additional [self study](#) using the recommended textbooks
and the on-line material is [essential](#)

Delivery of the module (1)

② Practicals

- Structure:

- 7 practicals with worksheets (3 Perl, 2 PHP, 2 JavaScript)

- gain understanding via practice

- get answers to questions about the lecture material

- U

- Sc

- <https://eduassistpro.github.io/>

- 1 pr

Practicals start in week 2

- Practicals assume familiarity with Linux and dep

- ~ To recap, use the worksheets available at

- cgi.csc.liv.ac.uk/~ullrich/COMP284/notes

- Practicals assume familiarity with the related lecture material

How to learn a new programming language

- Once you know how to program in one programming language, additional programming languages are best learned by a process of enquiry and practice guided by existing experience
- Typically, the questions that guide you are

- Wh

- Exa

- Wh

- Exa

- What happens if ...?

- Example: What happens if 1 is divided by 0?

- How do I ...?

- Example: How do I catch an exception?

- Talk to other people who are currently trying to learn the same language or have already learned it
 - Ask what has surprised them most

How to learn a new programming language

- Once you know how to program in one programming language, additional programming languages are best learned by a process of enquiry and practice

Assignment Project Exam Help

- The best kind of learning is learning by doing

~ Th

by p

- Work o

~ You need to convince employers that you have more substantive than 'toy' programs

~ The assignments are 'pretend' substantive p
but in reality are too small

- Employers value experience, in particular, the experience that you get from overcoming challenges
- ~ Assignments that are not challenging are of limited value

Add WeChat edu_assist_pr

Delivery of the module (3)

③ Office hours

Monday, 16:00 Ashton Room 1.03

but always arrange a meeting by e-mail first
(U.Hustadt@liverpool.ac.uk)

④ Ann

- Yes <https://eduassistpro.github.io>
- Always use your university e-mail account
if you want to contact me or any other member of staff

Add WeChat edu_assist_pro

Recommended texts

- Core reading

- R. Nixon:

Learning PHP, MySQL & JavaScript.
O'Reilly, 2009.

Harold Cohen Library: 518.561.N73 or e-book

Learning PHP . . . , 5th edition
O'Reilly, 2014.

- R. L. S

Lear
O'R
Har

Learning Perl, 7th edition.
O'Reilly, 2016.

- Further reading

- M. David:

HTML5: designing rich Internet applications
Focal Press, 2010.

Harold Cohen Library: 518.532.D24 or e-book

- N. C. Zakas:

Professional JavaScript for Web Developers.
Wiley, 2009.

Harold Cohen Library: 518.59.Z21 or e-book

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Assessment

- This is a coursework-based module
(no exam)

• Three assessment tasks need to be completed throughout the semester:

– Perl	Deadline: Friday,	2 March, 17:00
– PHP	Deadline: Mon	9
– Java	D	

- Effo <https://eduassistpro.github.io>
- Available at: <http://cgi.csc.liv.ac.uk/~ullrich/COMP284/>

Add WeChat edu_assist_pro

Attendance and Performance

	Students	Average Lecture Attendance	Average Practical Attendance	Average Module Mark
2011-12	33	76.0%	70.0%	63.1
2016-17	114	43.8%	38.3%	53.0

<https://eduassistpro.github.io>

- Add WeChat `edu_assist_pro`
- From 2014-15, screencasts of the lectures were available to students
- From 2015-16, the requirement to write a report on each program
- Hypothesis 1:
 $\text{Lecture Attendance} > 75\% \text{ and } \text{Practical Attendance} > 65\% \Leftrightarrow \text{Module Mark} > 62$
- Hypothesis 2:
 $\text{Screencasts Available} \Leftrightarrow \text{Module Mark} < 59$

Academic Integrity

- Plagiarism occurs when a student misrepresents, as his/her own work, the work, written or otherwise, of any other person (including another student) or of any institution.
- Collusion occurs where there is unauthorised co-operation between a student which involves:
- Fabrication of data in order to conceal a lack of legitimate data

If you are found to have plagiarised work, colluded with others or fabricated data, then you may fail COMP284.

Add WeChat `edu_assist_pro`

Serious 'offenders' may be excluded from the University

Do not try to take a 'shortcut'
You must do the work yourself!

Academic Integrity: Lab rules

- Do **not** ask another student to see any part of their code for a COMP284 assignment

→ contravention of this leads to **collusion**

Assignment Project Exam Help

- Do **not** show or make available any part of your code relating for a COM

→ co

- Do **not** share your code for a COMP284 assignment

→ contravention of this leads to **collusion**

- Lock your Lab PC when you leave it alone
- Where you use any material/code found on-line for an assignment, you **must** add comments to your code indicating its origin by a proper academic reference

→ contravention of this is **plagiarism**

→ acknowledged code re-use may still result in a lower mark

Add WeChat `edu_assist_pro`

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 2: Perl (Part 1)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

③ Perl: Overview

History

Applications

Java vs Perl

④ Scalar

De

Int

Stri

'Booleans'

Comparisons

⑤ Variables, Constants, and Assignments

Variables

Constants

Assignments

Variable interpolation

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Perl

- Originally developed by [Larry Wall](#) in 1987
Perl 6 was released in December 2015

Assignment Project Exam Help

imperative language with variables, expressions, assignment statements, blocks

- [Lisp](#) lists
- [https://eduassistpro.github.io](#)
- [AWK](#) (pattern scanning and processing language)
hashes / associative arrays, regular expressions
- [sed](#) (stream editor for filtering and transforming text)
regular expressions and substitution s///
- [Shell](#)
use of [sigils](#) to indicate [type](#) (\$ – scalar, @ – array, % – hash, & – procedure)
- [Object-oriented programming languages](#)
classes/packages, inheritance, methods

Add WeChat edu_assist_pro

Perl: Uses and applications

- Main application areas of Perl
 - text processing
 - easier and more powerful than sed or awk
 - system administration
 - easier and more powerful than shell scripts
- Other areas
 - web
 - cod
 - bioinformatics
 - linguistics
 - testing and quality assurance

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

Perl: Applications

- Applications written in Perl
 - Movable Type – web publishing platform
<http://www.movable-type.org/>
 - Request Tracker – issue tracking system
<http://bestpractical.com/rt/>
 - Slas – database-d
http://<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Perl: Applications

- Organisations using Perl
 - Amazon – online retailer
<http://www.amazon.co.uk>
 - BBC – TV/Radio/Online entertainment and journalism
<http://www.bbc.co.uk>
 - Boo
 - craigslist – clas
<http://www.craigslist.org>
 - IMDb – movie database
<http://www.imdb.com>
 - Monsanto – agriculture/biotech
<http://www.monsanto.co.uk/>
 - Slashdot – technology related news
<http://slashdot.org>

Assignment Project Exam Help

Add WeChat edu_assist_pro

Java versus Perl: Java

```
1 /* Author: Clare Dixon
2  * The HelloWorld class implements an application
3  * that prints out "Hello World".
4 */
5 public class HelloWorld {
6     // ---
7     /* Mai M
8     publ
9     s
10    }
11 }
```

Add WeChat edu_assist_pro

Edit-compile-run cycle:

- ① Edit and save as HelloWorld.java
- ② Compile using javac HelloWorld.java
- ③ Run using java HelloWorld

Java versus Perl: Perl

```
1 #!/usr/bin/perl
2 # Author: Ullrich Hustadt
3 # The Hello World script implements an application
4 # that prints out "Hello World".
5
6 print "He
```

Assignment Project Exam Help

Edit-run cycle

<https://eduassistpro.github.io/>

- ① Edit and save as `HelloWorld`
- ② Run using `perl HelloWorld`
This only needs to be done once!
- ③ Run using `./HelloWorld`

Perl

- Perl borrows features from a wide range of programming languages including imperative, object-oriented and functional languages

Assignment Project Exam Help

- Advantage: Programmers have a choice of programming styles
- Disad
- Perl m <https://eduassistpro.github.io/>
 - ~ Documenting and commenting Perl code is very important

Add WeChat edu_assist_pro

Perl

- Perl makes it easy to write completely incomprehensible code
 - ~ Documenting and commenting Perl code is very important

Assignment Project Exam Help

```
1  #!/usr/bin/perl
2  # Authors: Schwartz et al. / Ullrich Hustadt
3  # Text manip
4  #
5  # Reviewer
6  @lines = <PC>
7
8  # Go through the lines of the documentation, turn all text
9  # between single brackets to uppercase and remove the
10 # character in front of the opening angled bracket, then
11 # print the result
12 foreach (@lines) {
13     s/\w<([^\>]+)>/\U$1/g;
14     print;
15 }
```

<https://eduassistpro.github.io/>

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

In the example, there are more lines of comments than there are lines of code

Perl for Java programmers

- In the following we will consider various constructs of the Perl programming language
 - numbers, strings

- variables, constants
- assi
- con

<https://eduassistpro.github.io/>

- These will often be explained with reference to Java ('like Java', 'unlike Java')

Add WeChat edu_assist_pro

- Note that Perl predates Java
 - ~ common constructs are almost always inherited by both languages from the programming language C

Perl scripts

- A Perl script consists of one or more statements and comments
 - ~ there is no need for a main function (or classes)

Assignment Project Exam Help

- Whitespace before and in between statements is irrelevant

(Th

- Co

- Co

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Perl scripts

- Perl statements include

- Assignments

- Control structures

Assignment Project Exam Help

Every statement returns a value

- Perl da

- Scal

- Arr

- Hashes / Associative arrays

- Perl expressions are constructed from values and v

- operators and subroutines

- Perl expressions can have side-effects

- (evaluation of an expression can change the program state)

Every expression can be turned into a statement by adding a semi-colon

Scalar data

- A scalar is the simplest type of data in Perl
- A scalar is either

an integer number

0 2012 -40 1_263_978

- a float
- 1.2 2
- a string
'hello world' "hello world\n"

- Note: Add WeChat edu_assist_pr
- There is no 'integer type', 'string type' etc
- There are no boolean constants (true / false)

Integers and Floating-point numbers

- Perl provides a wide range of pre-defined mathematical functions

`abs(number)` absolute value

`log(number)` natural logarithm

`random(number)` random number between 0 and `number`

`sqrt(number)` square root

- Additi

`ceil`

`floo`

<https://eduassistpro.github.io/>

Note: There is no pre-defined round fu

```
use POSIX;  
print ceil(4.3); // prints '5'  
print floor(4.3); // prints '4'
```

- Remember: Floating-point arithmetic has its peculiarities

David Goldberg: What Every Computer Scientist Should Know About Floating-Point Arithmetic. Computing Surveys 23(1):5–48.

<http://perso.ens-lyon.fr/jean-michel.muller/goldberg.pdf>

Mathematical functions and Error handling

- Perl, PHP and JavaScript differ in the way they deal with applications of mathematical functions that do not produce a number

In Perl we have:

• `log(0)` produces an error message: Can't take log of 0

• `sqr`

• `1/0`

• `0/0`

<https://eduassistpro.github.io>

and execution of a script terminates when an error occurs.

- A possible way to perform `error handling` in Perl is as follows:

```
eval { ...run the code here...  
      1;  
} or do { ...handle the error here using $@... # catch  
};
```

The `special variable` `$@` contains the Perl syntax or routine error message from the last `eval`, `do-FILE`, or `require` command

Strings

Perl distinguishes between

- single-quoted strings and
- double-quoted strings

Assignment Project Exam Help

single-q

('taken lit

'hello

'don\'

' "hello"'

'backslash\\\'

'glass\\table'

'glass\table'

~> "hello"

~> backslash\

~> glass\table

~> glass\table

')

lit

n't

"\ "

"ba

"gl

"glass\table" ~> glass

"

ash\

table

Add WeChat edu_assist_pro

In Java, **single quotes** are used for single characters and
double quotes for strings

Double-quoted string backslash escapes

- In a single-quoted string \t is simply a string consisting of \ and t
- In a double-quoted string \t and other **backslash escapes** have the following meaning

Assignment Project Exam Help

Construct	Meaning
-----------	---------

\n

\f

\r

\t Tab

\l Lower case next letter

\L Lower case all following letters until \E

\u Upper case next letter

\U Upper case all following letters until \E

\Q Quote non-word characters by adding a backslash until \E

\E End \L, \U, \Q

<https://eduassistpro.github.io>

Add WeChat edu_assist_pr

UTF-8

- Perl supports **UTF-8** character encodings which give you access to non-ASCII characters
- The pragma

```
use utf8;
```

allows

- The fu

<https://eduassistpro.github.io/>

```
binmode(STDIN, ":encoding(UTF-8)");
binmode(STDOUT, ":encoding(UTF-8)");
```

ensures that UTF-8 characters are read correctly from STDIN and printed correctly to STDOUT

- The **Unicode::Normalize** module enables correct **decomposition** of strings containing UTF-8 encoded characters

```
use Unicode::Normalize;
```

UTF-8

Example:

```
binmode(STDOUT, ":utf8");
print "\x{4e14}\x{7d}\x{4e16}\x{754c}\n"; # chinese
print "\x{062a}\x{ref0}\n"; # arabic
```

For furth

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

String operators and automatic conversion

- Two basic operations on strings are

- string concatenation

"hello" . "world"

"hello" . , "world"

"\Uhello" . '\LWORLD'

"helelloworld"

'hello\u00d7world'

'HELLO\LWORLD'

- stri

"he

<https://eduassistpro.github.io>

- These operations can be combined

"hello" . "world" x 2

"

- Perl automatically converts between strings and numbers

2 . "worlds" ~> "2\u00d7worlds"

"2" * 3 ~> 6

2e-1 x 3 ~> "0.20.20.2" ("0.2" repeated three times)

"hello" * 3 ~> 0

'Booleans'

- Unlike Java, Perl does **not** have a boolean datatype
- Instead the values

```
0          # zero  
''         # empty string  
'0'  
undef  
()
```

<https://eduassistpro.github.io/>

all represent *false* while all other values repres

Add WeChat edu_assist_pro

'Boolean operators'

- Perl offers the same short-circuit boolean operators as Java: `&&`, `||`, `!`
Alternatively, `and`, `or`, `not` can be used

Assignment Project Exam Help

A	B	$(A \&\& B)$
<i>true</i>	<i>true</i>	<i>B (true)</i>
<i>true</i>	<i>false</i>	<i>false</i>
<i>false</i>	<i>true</i>	<i>false</i>
<i>false</i>	<i>false</i>	<i>false</i>

A	B	$(A B)$
<i>true</i>	<i>true</i>	<i>A (true)</i>
<i>true</i>	<i>false</i>	<i>true</i>
<i>false</i>	<i>true</i>	<i>true</i>
<i>false</i>	<i>false</i>	<i>false</i>

A	$(! A)$
<i>true</i>	<i>0 (false)</i>
<i>false</i>	<i>1 (true)</i>

Add WeChat edu_assist_pro

- Note that this means that `&&` and `||` are **not commutative**, that is, $(A \&\& B)$ is not the same as $(B \&\& A)$

```
($denom != 0) && ($num / $denom > 10)
```

Comparison operators

Perl distinguishes between **numeric comparison** and **string comparison**

Comparison	Numeric	String
Equal	<code>==</code>	<code>eq</code>
Not equal	<code>!=</code>	<code>ne</code>

<https://eduassistpro.github.io>

Greater than or equal to

`ge`

Examples

Add WeChat `edu_assist_pro`

```
35 == 35.0      # true
'35' eq '35.0'  # false
'35' == '35.0'  # true
35 < 35.0      # false
'35' lt '35.0'  # true
'ABC' eq "\Uabc" # true
```

Scalar variables

- Scalar variables start with \$ followed by a Perl identifier
- A Perl identifier consists of letters, digits, and underscores, but cannot start with a digit
Perl identifiers are case sensitive
- In Perl, a
- Scalar (there

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Scalar variables

- A **variable** also does **not** have to be **initialised** before it can be used, although **initialisation** is a good idea
- Uninitialised variables have the special value **undef**

Assignment Project Exam Help

However, **undef** acts
like 0

like ''
if an un

- To test whether a variable has value **un**

```
$s1 = "";  
print '$s1 eq undef: ', ($s1 eq undef) ? 'TRUE' : 'FALSE', "\n";  
print '$s1 defined: ', (defined($s1)) ? 'TRUE' : 'FALSE', "\n";  
print '$s2 defined: ', (defined($s2)) ? 'TRUE' : 'FALSE', "\n";  
  
$s1 eq undef: TRUE  
$s1 defined: TRUE  
$s2 defined: FALSE
```

Special Variables

- Perl has a lot of 'pre-defined' variables that have a particular meaning and serve a particular purpose

Variable	Explanation
\$_	The default or implicit variable
@_	
\$a, \$	
\$&	
\$/	input record separator, newline by default
\$\	output record separator, un
\$]	version of Perl used

- For a full list see

<https://perldoc.perl.org/perlvar.html#SPECIAL-VARIABLES>

Constants

Perl offers three different ways to declare `constants`

- Using the `constant` pragma:

```
use constant PI => 3.14159265359;
```

(A `pragma` is a module which influences some aspect of the compilation process)

- Using `Readonly`:

```
use Readonly;
Readonly $PI => 3.14159265359;
```

- Using the `Const::Fast` module:

```
use Const::Fast;
const $PI => 3.14159265359;
```

With our current Perl installation only `constant` works
→ variable interpolation with constants does not work

Assignments

- Just like Java, Perl uses the equality sign = for assignments:

```
$student_id = 200846369;  
$name = 'Jan Olsen';  
$student_id = "E00481370";
```

But no t

numb

<https://eduassistpro.github.io>

- An assi
namely (the final value of) the variable on the left
~ enables us to use an assignment as an expression

Add WeChat edu_assist_pro

Example:

```
$b = ($a = 0) + 1;  
# $a has value 0  
# $b has value 1
```

Binary assignments

There are also **binary assignment operators** that serve as **shortcuts** for arithmetic and string operations

Assignment Project Exam Help

Binary assignment	Equivalent assignment
<code>\$a += \$b</code>	<code>\$a = \$a + \$b</code>
<code>\$a %= \$b</code>	<code>\$a = \$a % \$b</code>
<code>\$a **= \$b</code>	<code>\$a = \$a ** \$b</code>
<code>\$a .= \$b</code>	<code>\$a = \$a . \$b</code>

<https://eduassistpro.github.io>
Add WeChat `edu_assist_pro`

Example:

```
# Convert Fahrenheit to Celsius:  
# Subtract 32, then multiply by 5, then divide by 9  
$temperature = 105;                      # temperature in Fahrenheit  
($temperature -= 32) *= 5/9;            # converted to Celsius
```

Variable declarations

- In Perl, variables can be declared using the `my` function
(Remember: This is not a requirement)

Assignment Project Exam Help

```
use strict;
```

enforc

other

<https://eduassistpro.github.io>

Exam

```
use strict;
$studentsOnCOMP284 = 153;
Global symbol "$studentOnCOMP284" requires explicit
    package name at ./script line 2.
Execution of ./script aborted due to compilation errors.
```

```
use strict;
my $studentsOnCOMP281;
$studentsOnCOMP281 = 154;
my $studentsOnCOMP283 = 53;
```

Variable interpolation

Variable interpolation

Any scalar variable name in a double quoted string
is automatically replaced by its current value

Assignment Project Exam Help

Example

```
$actor = 'Jeff'
$prize = "Acad
$year  = 2010;
print "1: $actor won the $prize in $year\n";
print "2: ", $actor, "won the ", $prize, "in ", $year, "\n";
```

Output:

```
1: Jeff Bridges won the Academy Award for Best Actor in 2010
2: Jeff Bridges won the Academy Award for Best Actor in 2010
```

Revision

Assignment Project Exam Help

Read

- Cha

of

<https://eduassistpro.github.io/>

R. L. Sch

Learning Perl.

O'Reilly, 2011

Add WeChat edu_assist_pro

Harold Cohen Library: 518.579.86.S39 or e-book

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 3: Perl (Part 2)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

⑥ Control structures

Conditional statements

Switch statements

While and Until-loops

For-loops

⑦ Lists and

Ide

List I

Contexts

List and array functions

Foreach-loops

Assignment Project Exam Help

⑧ Hashes

Identifiers

Basic hash operations

Foreach

Add WeChat edu_assist_pro

Control structures: conditional statements

The general format of **conditional statements** is very similar to that in Java:

```
if (condition) {  
    statements  
} elif (condition) {  
    sta  
} else {  
    sta  
}
```

<https://eduassistpro.github.io>

- *condition* is an arbitrary expression
- the *elif-clause* is optional and there can be more than one
- the *else-clause* is optional but there can be at most one
- in contrast to Java, the **curly brackets must be present** even if *statements* consist only of a single statement

Control structures: conditional statements

- Perl also offers two shorter conditional statements:

statement if (condition);

and

state

- In ana
Perl off

<https://eduassistpro.github.io/>

condition ? if_true_expr :

Examples: Add WeChat edu_assist_p

```
$descr = ($distance < 50) ? "near" : "far";
```

```
$size   = ($width < 10) ? "small" :
          ($width < 20) ? "medium" :
                           "large";
```

Blocks

- A sequence of statements in curly brackets is a **block**
 ~ an alternative definition of **conditional statements** is

Assignment Project Exam Help

```
if (condition) block  
elsif (condition) block  
else
```

- In **https://eduassistpro.github.io**

```
statement if (condition);  
statement unless (condition)
```

only a single statement is allowed
but **do block** counts as a single statement,
so we can write

```
do block if (condition);  
do block unless (condition);
```

Control structures: switch statement/expression

Starting with Perl 5.10 (released Dec 2007), the language includes a **switch statement** and corresponding **switch expression**

But these are considered **experimental** and need to be enabled explicitly

Assignment Project Exam Help

Example

```
use feature "switch";
given ($month) {
    when ([1,3,5,7,8,10,12]) { $days = 31 }
    when ([4,6,9,11]) { $days = 30 }
    when (2) { $days = 28 }
    default { $days = 0 }
}
```

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

Note: No explicit **break** statement is needed

Control structures: while- and until-loops

- Perl offers `while-loops` and `until-loops`

```
while (condition) {  
    statements  
}  
  
until (condition) {  
    statements  
}
```

Assignment Project Exam Help

- A 'proper' `until-loop` where the loop is executed at least once can be obtained as follows

```
do { statements } until (condition);
```

Add WeChat edu_assist_pro

The same construct also works for `if`,

In case there is only a single statement it is also possible to write

```
statement until (condition);
```

Again this also works for `if`, `unless` and `while`

Control structures: for-loops

- for-loops in Perl take the form

```
for (initialisation; test; increment) {  
    statements  
}
```

Again

consis

- Such a f

```
initialisation;  
while (test) {  
    statements;  
    increment;  
}
```

Add WeChat edu_assist_pro

Lists and Arrays

- A **list** is an ordered collection of scalars
- An **array (array variable)** is a variable that contains a list

Array variables start with @ followed by a Perl identifier

@*identifier*

An ar

- Perl **<https://eduassistpro.github.io>**

\$*identifier*[*index*]

to denote the element stored at position

The first array element has index 0

- Note that

\$*identifier*
@*identifier*

are two unrelated variables (but this situation should be avoided)

List literals

- A **list** can be specified by a **list literal**, a comma-separated list of values enclosed by parentheses

```
(1, 2, 3)
("adam", "ben", "colin", "david")
("adam", 1, "ben", 3)
()
(1..10, 1)
($start,
```

<https://eduassistpro.github.io>

- List literals** can be assigned to an **array**:

```
@numbers = (1..10, 15, 20..30);
@names = ("adam", "ben", "colin", "david")
```

- Examples of more complex assignments, involving arrays:

```
@numbers = (1..10, undef, @numbers, ());
@names = (@names, @numbers);
```

- Note that arrays do **not** have a pre-defined size/length

Size of an array

- There are three different ways to determine the size of an array

```
$arraySize = scalar(@array);  
$arraySize = @array;  
$arraySize = $#array + 1;
```

Assignment Project Exam Help

- One ca
in the ra
- But Pe

The expression `$array[-index]`

is equivalent to `$array[scalar(@array)-1]`

Add WeChat edu_assist_pr

Example:

`$array[-1]` is the same as `$array[scalar(@array)-1]`

is the same as `$array[$#array]`

that is the last element in `@array`

Array index out of bound

- Perl, in contrast to Java, allows you to access array indices that are **out of bounds**
- The value `undef` will be returned in such a case

Assignment Project Exam Help

```
@array = (0, undef, 22, 33);  
print '$arra
```

```
print '$arra
```

```
$array[1] = , which IS undef  
$array[5] = , which IS undef
```

Add WeChat edu_assist_p

- The function `exists` can be used to determine if an index is within bounds and has a value (including `undef`) associated with it

```
print '$array[1] exists:', exists($array[1]) ? "T": "F", "\n";  
print '$array[5] exists:', exists($array[5]) ? "T": "F", "\n";  
$array[1] exists: T  
$array[5] exists: F
```

Scalar context versus list context

- Scalar context

when an expression is used as an argument of an operation that requires a scalar value, the expression will be evaluated in a scalar context

Assignment Project Exam Help

Example:

```
$array
```

~ @a

in a s

array

<https://eduassistpro.github.io>

- List context

when an expression is used as an argument of an operation that requires a list value, the expression will be evaluated in a list context

a list value, the expression will be evaluated in a list context

Example:

```
@sorted = sort 5;
```

~ A single scalar value is treated as a list with one element in a list context

Scalar context versus list context

Expressions behave differently in different contexts following these rules:

- Some operators and functions automatically return different values in different contexts.

```
$line  = <IN>;      # return one line from IN  
@lines = <I
```

<https://eduassistpro.github.io>

- If an expression returns a **list value** in a **scalar context**, then by default Perl will convert it into a scalar value by taking the last element of the returned list value

Add WeChat edu_assist_pro

List functions

Function	Semantics
<code>grep(<i>expr</i>, <i>list</i>)</code>	in a list context, returns those elements of <i>list</i> for which <i>expr</i> is true; in a scalar context, returns the number of
<code>join(<i>list</i>)</code>	ents ratio
<code>reverse(<i>list</i>)</code>	returns a lis
<code>sort(<i>list</i>)</code>	returns a list wi standard s
<code>split(/<i>regexp</i>/, <i>string</i>)</code>	returns a list obtained by splitting <i>string</i> into substring using <i>regexp</i> as separator
<code>(<i>list</i>) x <i>number</i></code>	returns a list composed of <i>number</i> copies of <i>list</i>

Array functions: push, pop, shift, unshift

Perl has no `stack` or `queue` data structures,
but has `stack` and `queue` functions for `arrays`:

Function	Semantics
<code>push(@array1, value)</code> <code>push(</code>	appends an element or an entire list to the
<code>pop(@array1)</code>	extracts the last element from an array and returns it
<code>shift(@array1)</code>	<small>shift extracts</small> and returns it
<code>unshift(@array1, value)</code> <code>unshift(@array1, list)</code>	insert an element or an entire list at the start of an array variable; returns the number of elements in the resulting array

Add WeChat `edu_assist_pro`

Array operators: push, pop, shift, unshift

Example:

```
1 @planets = ("earth");
2 $shift (@planets, "mercury", "venus");
3 push(@planets, "mars", "jupiter", "saturn");
4 print "A
5 $last = pop(@planets);
6 print "L
7 $first = shift(@planets);
8 print "Array@1: ", join(" ", @planets), "\n";
9 print "Array@2: ", $first, " ", $last, "\n";
```

Output:

```
Array@1: mercury venus earth mars jupiter saturn
Array@2: mercury venus earth mars jupiter
Array@3: venus earth mars jupiter
          @4: mercury saturn
```

Array operators: delete

- It is possible to **delete** array elements

- `delete($array[index])`

- removes the value stored at `index` in `@array` and returns it

- only if `index` equals `$#array` will the array's size shrink to the position `ts()`

```
@array = (0, 11, 22, 33)
delete($array[2])
print '$array [2] exists: ', exists($array[2])?"T":"F", "\n";
print 'Size of $array: ', $#array+1
delete($array[3])
print '$array [3] exists: ', exists($array[3])?"T":"F", "\n";
print 'Size of $array: ', $#array+1
```

```
$array [2] exists: F
Size of $array: 4
$array [3] exists: F
Size of $array: 2
```

Control structures: foreach-loop

Perl provides the `foreach`-construct to 'loop' through the elements of a list

```
foreach $variable (list) {  
    statements  
}
```

where \$

list in each t

<https://eduassistpro.github.io/>

Example:

```
@my_list = (1..5,20,11..18);  
foreach $number (@my_list) {  
    $max = $number if (!defined($max) || $number > $max);  
}  
print("Maximum number in ", join(' ', @my_list), " is $max\n");
```

Output:

```
Maximum number in 1,2,3,4,5,20,11,12,13,14,15,16,17,18 is 20
```

Control structures: foreach-loop

Changing the value of the **foreach-variable** changes the element of the list that it currently stores

Assignment Project Exam Help

Example:

```
@my_list = (1..5,20,11..18);
print "Before: @my_list\n";
foreach $num (@my_list) {
    $num = $num + 1;
}
print "After: @my_list\n";
```

Output: Add WeChat edu_assist_pro

```
Before: 1, 2, 3, 4, 5, 20, 11, 12, 13, 14, 15, 16, 17, 18
After: 2, 3, 4, 5, 6, 21, 12, 13, 14, 15, 16, 17, 18, 19
```

Note: If no variable is specified, then the special variable `$_` will be used to store the array elements

Control structures: foreach-loop

An alternative way to traverse an array is

```
foreach $index (0..$#array) {  
    statements  
}
```

Assignment Project Exam Help

where an el

statement

`$index]` in

Example

```
@my_list = (1..5,20,11..18);  
foreach $index (0..$#my_list) {  
    $max = $my_list[$index] if ($my_list[$index] > $max);  
}  
print("Maximum number in ", join(' ', @my_list), " is $max\n");
```

Add WeChat edu_assist_pro

Control structures: foreach-loop

- In analogy to `while`- and `until`-loops, there are the following variants of `foreach`-loops:

`do { statements } foreach list;`

`statement foreach list;`

In the ex

`$_` wil

<https://eduassistpro.github.io>

- Instead

`do { statements } for list`

`statements for list,`

Example:

```
print "Hello $_!\n" foreach ("Peter", "Paul",  
                           "Mary");
```

Control structures: last and next

- The `last` command can be used in while-, until-, and foreach-loops and discontinues the execution of a loop

```
while ($value = shift(@ta)) {  
    $written = print(FILE $value);  
    if (! $written) { last; }  
}  
  
# Executi
```

- The `next` command exits the body of a loop and moves the execution to the next iteration

```
foreach $x (2..2) {  
    if ($x == 0) { next; }  
    printf("10 / %2d = %3d\n", $x, (10/$x));  
}  
  
10 / -2 = -5  
10 / -1 = -10  
10 / 1 = 10  
10 / 2 = 5
```

Add WeChat edu_assist_pro

Hashes

- A hash is a **data structure** similar to an **array** but it associates scalars with a **string** instead of a number
- Alternatively, a hash can be seen as a **partial function** mapping **strings** to **scalars**
- Reme

• Hash v <https://eduassistpro.github.io/>

%identifier

A **hash variable** denotes the entirety of the hash

- Perl uses

\$identifier{key}

where **key** is a **string**, to refer to the value associated with **key**

Hashes

- Note that

Assignment Project Exam Help

are two unrelated variables (but this situation should be avoided)

- An easy way to dump a hash is the following:

```
use Data::Dumper;  
$Data::Dumper::Terse = 1;  
print Dumper \%hash;
```

Add WeChat edu_assist_pro

Note the use of `\%hash` instead of `%ha`

(`\%hash` is a reference to `%hash`)

Data::Dumper can produce string representations for arbitrary Perl data structures

Basic hash operations

- Initialise a hash using a list of key-value pairs

`%hash = (key1, value1, key2, value2, ...);`

Assignment Project Exam Help

- Initialise a hash using a list in big arrow notation

`%hash ..);`

- Assoc <https://eduassistpro.github.io>

`$hash{key} = value;`

- Remember that `undef` is a scalar value

`$hash{key} = undef;`

extends a hash with another key but unknown value

Basic hash operations

- One can use the `exists` or `defined` function to check whether a key exists in a hash:

`exists $hash{key}`

Note that if `$hash{key}` eq `undef`, then `exists $hash{key}` is true

- The `delete` function removes a key from a hash:

`delete ($hash{key});`

After executing `delete ($hash{key});`,
`exists $hash{key}` will be
false

Add WeChat edu_assist_pro

- The `undef` function removes the contents and memory allocated to a hash:

`undef %hash`

Basic hash operations

- It is also possible to assign one hash to another

```
%hash1 = %hash2;
```

In contrast to C or Java this operation creates a copy of %hash1 that is then assigned to %hash1

Exam

<https://eduassistpro.github.io>

```
%hash1 = ();
%hash2 = %hash1;
$hash1{'b'} = 4;
print "\$hash1{'b'} = \$hash1{'b'}\n";
print "\$hash2{'b'} = \$hash2{'b'}\n";
```

Add WeChat edu_assist_pro.com

Output:

```
$hash1{'b'} = 4
$hash2{'b'} = 2
```

The each, keys, and values functions

<code>each %hash</code>	returns a 2-element list consisting of the key and value for the next element of <code>%hash</code> , so that one can iterate over it
<code>values %hash</code>	returns a list consisting of all the values of <code>%hash</code> ,
<code>keys %hash</code>	https://eduassistpro.github.io

Examples:

```
while (my ($key, $value) = each %hash)
    statements
}
```

```
foreach $key (sort keys %hash) {
    $value = $hash{$key};
}
```

Example: Two-dimensional hash as a 'database'

```
1 use List::Util "sum";
2 $name{ '200846369' } = 'Jan\ Olson';
3 $marks{ '200846369' }{ 'COMP201' } = 61;
4 $marks{ '200846369' }{ 'COMP207' } = 57;
5 $mark
6 $mark
7
8 $average = sum(values($marks{ '200846369' }))/
9     scalar(values($marks{
10 print("avg: $average\n");
```

Output:

avg: 60

Example: Frequency of words

```
1 # Establish the frequency of words in a string
2 $string = "peter\u00a9paul\u00a9mary\u00a9paul\u00a9jim\u00a9mary\u00a9paul";
3
4 # Split the string into words and use a hash
5 # to accum
6 ++$c;
7
8 # Print the frequency of each word found in the
9 # string
10 while((($key,$value) = each %count) > 0)
11   print("$key \u00a9=> \u00a9$value;\u00a9");
12 }
```

Assignment Project Exam Help
<https://eduassistpro.github.io>

Output:

```
jim => 1; peter => 1; mary => 2; paul => 3
```

Revision

Assignment Project Exam Help

- Chapter 3: Lists and Arrays
- Cha

of <https://eduassistpro.github.io/>

R. L. Schwartz, brian d foy, T. Phoenix:

Learning Perl:
Add WeChat edu_assist_pro
O'Reilly, 2011.

Harold Cohen Library: 518.579.86.S39 or e-book

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 4: Perl (Part 3)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Assignment Project Exam Help

⑨ Regular Expressions

Intr

Ch

Character classes

Quantifiers

Add WeChat edu_assist_pro

Regular expressions: Motivation

Suppose you are testing the performance of a new sorting algorithm by measuring its runtime on randomly generated arrays of numbers of a given length:

Assignment Project Exam Help

Generating an unsorted array with 10000 elements took 1.250 seconds

Sortin

Genera

Sortin

Genera

Sorting took 8.951 seconds

<https://eduassistpro.github.io>

Your task is to write a program that determines the average runtime of the sorting algorithm:

Add WeChat edu_assist_pro

Average runtime for 10000 elements is 8.886 seconds

Solution: The regular expression `/^Sorting took (\d+\.\d+) seconds/` allows us to get the required information

~ Regular expressions are useful for information extraction

Regular expressions: Motivation

Suppose you have recently taken over responsibility for a company's website. You note that their HTML files contain a large number of URLs containing superfluous occurrences of '...'. e.g.

Assignment Project Exam Help
`http://www.myorg.co.uk/info/refund/.../vat.html`

Your task is
equivalent to

https://eduassistpro.github.io

while making sure that relative URLs like

`.../video/display.html` **Add WeChat edu_assist_pro**

are preserved

Solution: `s!/[^\\/]++/\\.\\.!!;` removes a superfluous dot-segment

~ Substitution of regular expressions is useful for text manipulation

Regular expressions: Introductory example

```
\Ahttps?:\/\/[^\/]+\/.\w+\/(cat|dog)\/\1
```

Assignment Project Exam Help

- h, t, p, s, :, \/, c, a, t, d, o, g are characters
- ? and
- [^\/] https://eduassistpro.github.io
- . is a me
- (cat|dog) is alternation within a capture group
- \1 is a backreference to a capture group

Add WeChat edu_assist_pr

Pattern match operation

- To match a regular expression *regeexpr* against the special variable `$_` simply use one of the expressions `/regeexpr/` or `m/regeexpr/`

This is called a pattern match
• `$_` is the target string of the pattern match

- In a `scal` dependent

`''`)

<https://eduassistpro.github.io>

```
if (/^https  
    ... }
```

```
if (m/^https://[^/]+/[wW](cat|dog)/1/)  
    ... }
```

Add WeChat `edu_assist_pro`

Regular expressions: Characters

The simplest regular expression just consists of a sequence of

- alphanumeric characters and
- non-alphanumeric characters escaped by a backslash

that matches exactly this sequence of characters occurring as a substring in the targ

<https://eduassistpro.github.io/>

```
$_ = "ababcbcd"  
if (/cbc/) { print "Match\n"} else { print "No match\n" }
```

Output:

Match

Add WeChat edu_assist_pr

```
$_ = "ababcbcdcde";  
if (/dbd/) { print "Match\n"} else { print "No match\n" }
```

Output:

No match

Regular expressions: Special variables

- Often we do not just want to know whether a regular expression matches a target string, but retrieve additional information
- The `\$-[0]` can be used to retrieve the start position of the match

Note th

- The `sp` after th
- The `\$&` returns the match it

```
$_ = "abababbc;dcde";  
if (/cbc/) { print "Match found at position $-[0]: $&\n"}  
Add WeChat edu_assist_pr
```

Output:

```
Match found at position 4: cbc
```

Regular expressions: Special escapes

There are various **special escapes** and **metacharacters** that match more than one character:

Assignment Project Exam Help	
.	Matches any character except '\n'
\w	Matches a 'word' character (alphanumeric)
\W	
\s	Match a whitespace
\S	Match a non-whitespace
\d	Match a decimal digit
\D	Match a non-digit
\p{ <i>UnicodeProperty</i> }	Match <i>UnicodeProperty</i> characters
\P{ <i>UnicodeProperty</i> }	Match non- <i>UnicodeProperty</i> characters

Regular expressions: Unicode properties

- Each **unicode character** has one or more **properties**, for example, which script it belongs to

\P{Script} matches all characters that have a particular property

- \P{U}

- Exam

<https://eduassistpro.github.io/>

Ara	
ASCII	ASCII characters
Currency_Symbol	Currency symbols
Digit	Digits in all scripts
Greek	Greek characters
Han	Chinese kanxi or Japanese kanji characters
Space	Whitespace characters

Add WeChat edu_assist_pr

See <http://perldoc.perl.org/perluniprops.html> for a complete list

Regular expressions: Character class

- A **character class**, a list of characters, special escapes, metacharacters and unicode properties enclosed in square brackets, matches any **single character** from within the class, for example, `[ad\t\n\-\\\09]`

- One m
for exa
- A **caret**

that is, it matches any **single character** that is **not** from within the class, for example, `[^01a-z]`

Add WeChat edu_assist_p

```
$_ = "ababcbcdcde";  
if (/[^bcd][bc-e][^bcd]/) {  
    print "Match at positions $-[0] to ",$+[0]-1,": $&\n";}
```

Output:

```
Match at positions 8 to 10: cde
```

Quantifiers

- The constructs for regular expressions that we have so far are not sufficient to match, for example, natural numbers of arbitrary size
- Also, writing a regular expression for, say, a nine digit number would be tedious

Assignment Project Exam Help

This is ma

<i>regex</i>	https://eduassistpro.github.io/
<i>regexp⁺</i>	Match <i>regexp</i> 1 or more times
<i>regexp?</i>	Match <i>regexp</i> 1 or 0 times
<i>regexp{1}</i>	Match <i>regexp</i> exactly once
<i>regexp{n}</i>	Match <i>regexp</i> at least n
<i>regexp{n,m}</i>	Match <i>regexp</i> at least n but not more than m times

Quantifiers are greedy by default and match the longest leftmost sequence of characters possible

Quantifiers

<i>regexp*</i>	Match <i>regexp</i> 0 or more times
<i>regexp+</i>	Match <i>regexp</i> 1 or more times
<i>regexp?</i>	Match <i>regexp</i> 1 or 0 times
<i>regexp{n}</i>	Match <i>regexp</i> exactly n times
<i>regex</i>	
<i>regex</i>	

Example

<https://eduassistpro.github.io>

```
$_ = "Sorting\u took\u 10.486\u seconds";
if (/d+\.\d+)/ {
    print "Match\u at\u position\u $-[0]\u to\u $+[0]-1";
}
$_ = "E00481370";
if (/[A-Z]0{2}(\d+)/) {
    print "Match\u at\u positions\u $-[1]\u to\u ",$+[1]-1, ":\u $1\n";
}
```

Output:

```
Match at positions 13 to 18: 10.486
Match at positions 3 to 8: 481370
```

Quantifiers

Example:

```
$_ = "E00481370";  
if ((\d+){  
    print "Match at positions ${[0]} to ${[0]}, ${\$&}\n",
```

Assignment Project Exam Help

Output:

```
Match at posit
```

<https://eduassistpro.github.io>

- The regular expression
- As the example illustrates, the regular expression
 - matches as early as possible
 - matches as many digits as possible
 - ~ quantifiers are greedy by default

Add WeChat edu_assist_pro

Revision

Read

- Chapter 7: In the World of Regular Expressions
- Chapter 8: Matching with Regular Expressions

of

R. L. Sch

Learning Perl.

O'Reilly 2011

Add WeChat edu_assist_pr

- <http://perldoc.perl.org/perlre.html>
- <http://perldoc.perl.org/perlretut.html>
- <http://www.perlfest.com/articles/regextutor.shtml>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 5: Perl (Part 4)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Assignment Project Exam Help

⑩ Regular expressions

Character classes

Alternation

Assertions

Modifiers

Binding operators

Add WeChat edu_assist_pro

Regular expressions: Capture groups and backreferences

- We often encounter situations where we want to identify the repetition of the same or similar text, for example, in HTML markup:

Assignment Project Exam Help

```
<strong>...</strong>  
<li> ... </li>
```

- We might want to capture the bolded text, but then lose the list items.
- We can capture both by using regular expressions:

```
<strong>.*</strong>  
<li>.*</li>
```

Add WeChat edu_assist_pro

but we cannot characterise both without losing fidelity, for example:

```
<\w+>.*<\/\w+>
```

does not capture the 'pairing' of HTML tags

Regular expressions: Capture groups

The solution are capture groups and backreferences

(<i>resexpr</i>)	creates a capture group
(<i><name></i>) <i>resexpr</i>	creates a named capture group
(?: <i>resexpr</i>)	creates a non-capturing group
\N, \g	
\g{ <i>na</i> }	

<https://eduassistpro.github.io>

Examples:

- 1 /Sorting took (\d+.\d+) seconds/
- 2 /<(\w+)>.*<\/\1>/
- 3 /([A-Z])0{2}(\d+)/
- 4 /(?(?<c1>\w)(?<c2>\w)\g{c2})\g{c1}/
- 5 /((?(?<c1>\w)(?<c2>\w)\g{c2})\g{c1})/

Regular expressions: Capture groups

Via **capture variables** the strings matched by a **capture group** are also available outside the pattern in which they are contained

Assignment Project Exam Help

\$1	string matched by capture group (where <i>N</i> is a natural number)
$\$+\{na$	

The mate code bloc

<https://eduassistpro.github.io>

Example:

```
$_ = "abba_dabba_doo"
if (/((?<c1>\w)(?<c2>\w)\g{c2}\g{c1})/) {
    print "Match found: $1\n" }
```

Output:

```
Match found: abba
```

Regular expressions: Alternations

- The regular expression $regexp1 | regexp2$ matches if either $regexp1$ or $regexp2$ matches

This type of regular expression is called an **alternation**.

- Within a larger regular expression we need to enclose alternations in a **capt**

(^{reg}

<https://eduassistpro.github.io>

Examples:

1 /Mr | Ms | Mrs | Dr/

2 /cat | dog | bird/

3 /(?:Bill | Hillary) Clinton/

Add WeChat edu_assist_pro

Regular expressions: Alternations

- The **order of expressions** in an **alternation** only matters if one expression matches a sub-expression of another

Assignment Project Exam Help

Example:

```
1 $_ = "cats
2 if (/ca/
3 if (/do/
4 if (/(\d{3})/ { print "Match 1: $1\n" }
5 if (/(\d{3})/) { print "Match 2: $1\n" }
```

Output:

Add WeChat edu_assist_pro

```
Match 1: cat
Match 2: cat
Match 3: dog
Match 4: dogs
```

Regular expressions: Anchors

Anchors allow us to fix where a match has to start or end

\A	Match only at string start
\z	Match only at string start (default)
\z	Match only at a line start (in <code>/m</code>)
\\$	
\b	Match word boundary (between \w
\B	Match except at word boundary

Example:

```
$_ = "The\u00a0girl\u00a0who\nplayed\u00a0with\u00a0fire\n";
if (/fire\z/) { print "'fire'\u00a0at\u00a0string\u00a0end\n" }
if (/fire\Z/) { print "'fire'\u00a0at\u00a0string\u00a0end\u00a0modulo\u00a0\\n\\n" }

'fire'\u00a0at\u00a0string\u00a0end\u00a0modulo\u00a0\\n
```

Regular expressions: Modifiers

Modifiers change the interpretation of certain characters in a regular expression or the way in which Perl finds a match for a regular expression

Assignment Project Exam Help

/ / Default	<p>'.' matches any character except '\n'</p>
/ /s	<p>'.' matches any character including ' ' '^' matches only at string start '\$' matches only at string end modulo piece</p>
/ /m	<p>Treat string as a set of multiple lines '.' matches any character except '\n' '^' matches at a line start '\$' matches at a line end</p>

Regular expressions: Modifiers

Modifiers change the interpretation of certain characters in a regular expression or the way in which Perl finds a match for a regular expression

Assignment Project Exam Help

/ /sm Treat string as a single long line, but detect multiple lines
'.' matches any character including '\n'

/ /i

<https://eduassistpro.github.io>

Example:

Add WeChat edu_assist_pr
\$_ = "bill\\nClinton";
if ((Bill|Hillary).Clinton)/smi) { print "Match:\$1\\n" }

Output:

Match: bill

Clinton

Regular expressions: Modifiers (/ /g and / /c)

Often we want to process all matches for a regular expression, but the following code has not the desired effect

```
$_ = "1223";  
while (/d+/) { print "Match starts at $-[0]: $&\n" }
```

The code a

```
Match starts a 0
```

<https://eduassistpro.github.io>

To obtain

the / /g modifier:

/ /g	In <i>scalar context</i> , successive invocations always move from match to match, keeping track of the string
	In <i>list context</i> , returns a list of matched capture groups, or if there are no capture groups, a list of matches to the whole regular expression

Regular expressions: Modifiers (/ /g and / /c)

With the / /g modifier our code works as desired:

```
$_ = "11|22|33";
while (/^\d+/g) { print "Match starts at ", $-[0], "\n" }
```

Output:

```
Match starts at 0
Match starts at 3
Match starts at 6
```

<https://eduassistpro.github.io>

An example in a list context is the following:

```
$_ = "ab|11|cd|22|ef|33";
@numbers = (/^\d+/g);
print "Numbers: ", join(" | ", @numbers), "\n";
```

Output:

```
Numbers: 11 | 22 | 33
```

Read / /g as: Start to look for a match from the position where the last match using / /g ended

Regular expressions: Modifiers (/ /g and / /c)

The **current position** in a string for a regular expression *regexp* is associated with the string, not *regexp*

→ different regular expressions for the same strings will move forward the same position when used with / /g

~ differ

move f

Example

```
$_ = "ab\u00d711\u00d7cd\u00d722\u00d7ef\u00d733";
if (/d+/g) { print "Match starts at $-[0]: $&\n" }
if (/a-z/g) { print "Match starts at $-[0]: $&\n" }
if (/d+/g) { print "Match starts at $-[0]: $&\n" }
```

Output:

```
Match starts at 3: 11
Match starts at 6: cd
Match starts at 9: 22
```

Regular expressions: Modifiers (/ /g and / /c)

A failed match or changing the target string resets the position

```
1 $_ = "ab\u0011cd\u0022ef\u0033";
2 if (/^c+/g) { print "2: Match starts at $-[0]: $&\n" }
3 if (/d/g) { print "3: Match starts at $-[0]: $&\n" }
4 if (/d+/g) { print "4: Match starts at $-[0]: $&\n" }
```

Output:

```
2: Match starts at a
4: Match starts at a
```

To prevent the reset, an additional modifier

```
1 $_ = "ab\u0011cd\u0022ef\u0033";
2 if (/d/gc) { print "2: Match starts at $-[0]: $&\n" }
3 if (/ab/gc) { print "3: Match starts at $-[0]: $&\n" }
4 if (/d/g) { print "4: Match starts at $-[0]: $&\n" }
```

Output:

```
2: Match starts at 3: 11
4: Match starts at 9: 22
```

Generating regular expressions on-the-fly

The Perl parser will expand occurrences of `$variable` and `@variable` in regular expressions

→ regular expressions can be constructed at runtime

Example:

```
$_ = "Bart\u00eatea
@keywords = ();
while ($keyword) {
    print "Match found for $keyword: $&\n" if /$keyword/i;
}
```

Output:

Add WeChat edu_assist_pr

```
Match found for bart: Bart
Match found for lisa: Lisa
Match found for L\w+: Lisa
Match found for t\w+: teases
```

Binding operator

Perl offers two **binding operators** for regular expressions

`string =~ /regexp/` true iff `regexp` matches `string`

`string !~ /regexp/` true if `regexp` does not match `string`

Assignment Project Exam Help

- Note th
- Most o
return t
variab

<https://eduassistpro.github.io>

Example:

Add WeChat edu_assist_pro

```
$name = "Ullrich Hustadt";
if ($name =~ /(Mr|Ms|Mrs|Dr)?\s*(\w+)/) {print "Hello $1\n";
$name = "Dave Shield";
if ($name =~ /(Mr|Ms|Mrs|Dr)?\s*(\w+)/) {print "Hello $2\n"}
```

Hello Ullrich

Hello Dave

Pattern matching in a list context

- When a pattern match /*regexp*/ is used in a list context, then the return value is

- a list of the string matched by the capture groups in *regexp* if the match succeeds and *regexp* contains capture groups, or

- (a list
if the
- an e

<https://eduassistpro.github.io>

```
$name = "Dr\u00a0Ullrich\u00a0Hustadt";
($t,$f,$1) = ($name =~ /(Mr|Ms|Mrs|Dr)?\u00a0(.+)\u00a0(.+)/);
print "Name:\u00a0$t,\u00a0$f,\u00a0$1\n";
$name = "Dave\u00a0Shield";
($t,$f,$1) = ($name =~ /(Mr|Ms|Mrs|Dr)?\u00a0(.+)\u00a0(.+)/);
print "Name:\u00a0$t,\u00a0$f,\u00a0$1\n";
```

Output:

```
Name: Dr, Ullrich, Hustadt
Name: , Dave, Shield
```

Pattern matching in a list context

- When a pattern match `/regexp/g` is used in a `list` context, then the return value is

a list of the string matched by the capture groups in `/regexp` each time regex matches

provided that `regexp` contains capture groups, or

- a list c
mat
- an e

`https://eduassistpro.github.i`

```
$string = "firefox: 10.3 seconds; chrome: 9.5 seconds";
%performance = ($string =~ /\w+\:\s+\d+\.\d+\s+seconds/sg);
foreach $system (keys %performance) {
    print "$system -> $performance{$system}\n" }
```

Output:

```
firefox -> 10.3
chrome -> 9.5
```

Revision

Read

- Chapter 7: In the World of Regular Expressions
- Chapter 8: Matching with Regular Expressions

of

R. L. Sch

Learning Perl.

O'Reilly 2011

Add WeChat edu_assist_pr

- <http://perldoc.perl.org/perlre.html>
- <http://perldoc.perl.org/perlretut.html>
- <http://www.perlfest.com/articles/regextutor.shtml>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 6: Perl (Part 5)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

Assignment Project Exam Help

⑪ Substitution

Binding operators

Capture variables

Mo

⑫ Subro

<https://eduassistpro.github.io>

Introduction

Defining a subroutine

Parameters and arguments

Calling a subroutine

Persistent variables

Nested subroutine definitions

Add WeChat edu_assist_pr

Substitutions

`s/regexp/replacement/`

- Searches a variable for a match for *regexp*, and if found, replaces that match with a string specified by *replacement*
- In both **scalar context** and **list context** returns the number of substitutions made
- If no variables are specified, the substitution is applied to the specified scalar variable
- The **binding operator** `!~` only negates the regular expression, it does not affect the manipulation of the text

Add WeChat edu_assist_pro

The delimiter / can be replaced by some other pair of characters, for example:

`s!regexp! replacement!!` or `s<regexp>[replacement]`

Substitutions

Example:

```
$text = "http://www.myorg.co.uk/info/refund/..vat.html";
$text =~ s!/[^\>]*/! . !;
print "$text\n";
```

Assignment Project Exam Help

Output:

```
http://ww
```

<https://eduassistpro.github.io>

Example

```
$_ = "Yabba_dabba_doo";
s/bb/dd/;
print $_, "\n";
```

Add WeChat edu_assist_p

Output:

```
Yadda dabba doo
```

Note: Only the first match is replaced

Substitutions: Capture variables

`s/regexp/replacement/`

- Perl treats *replacement* like a double-quoted string
 - ↳ Backslash escapes work as in a double-quoted string

Assignment Project Exam Help

\	
\\	
\u	Upper case next letter
\U	Upper case all following letters

~ variable interpolation is applied, including `ca`

<code>\$N</code>	string matched by capture group <i>N</i> (where <i>N</i> is a natural number)
<code>+\${name}</code>	string matched by a named capture group

Substitutions: Capture variables

Example:

```
$name = "Dr\u00a0Ullrich\u00a0Hustadt";  
$name =~ s/(Mr|Ms|Mrs|Dr)\?|\s+(\w+)\s+(\w+)/\u00d6$2\u00d7E, $2/;  
print "$name\n";
```

```
$name = "Dave\u00a0  
$name =  
print "$name";
```

Output:

```
HUSTADT, Ullrich  
SHIELD, Dave
```

Add WeChat edu_assist_pro

Substitutions: Modifiers

Modifiers for substitutions include the following:

s/ / /g	Match and replace globally , that is, all occurrences
s/ / /i	Case-insensitive pattern matching
s/ / /m	Treat string as multiple lines
s/ / /s	
s/ / /e	

Combin

<https://eduassistpro.github.io>

Example:

```
$_ = "Yadda adda dada doo";
s/bb/dd/g;
print $_, "\n";
```

Output:

Yadda dadda doo

Substitutions: Modifiers

Modifiers for substitutions include the following:

s/ / ./e Evaluate the right side as an expression

Example:

```
1 $text = "The
2 $text =
3
4 print "$t
5 $text =~ s!(\d+\.\d+)!sprintf("%d", $1+0.5)!e;
6 print "$text\n";
```

The temperature is 40.555555555555 degrees Celsius
The temperature is 41 degrees Celsius

Better:

```
1 $text = "The temperature is 105 degrees Fahrenheit";
2 $text =~ s!(\d+) degrees Fahrenheit!
3
4         sprintf("%d", ((\$1-32)*5/9)+0.5).
5         " degrees Celsius"!e;
```

Regular Expressions and the Chomsky Hierarchy

- In Computer Science, formal languages are categorised according to the type of grammar needed to generate them (or the type of a

- recog Perl re at least recognise all context-free languages

- However, this does not mean regular expression's parsing context-free languages
- Instead there are packages specifically for parsing context-free languages or dealing with specific languages, e.g. HTML, CSV

Assignment Project Exam Help

Add WeChat ^{Cho} edu_assist_pr

Java methods versus Perl subroutines

- Java uses **methods** as a means to encapsulate sequences of instructions
- In **Java** you are expected
 - to declare the **type** of the **return value** of a method
- to provide a list of parameters, each with a distinct name, and t

<https://eduassistpro.github.io/>

```
public static int sum(int f, int s) {  
    f = f+s;  
    return f;  
}  
  
public static void main(String[] args) {  
    System.out.println("Sum of 3 and 4 is " + sum(3, 4))  
}
```

Add WeChat **edu_assist_pro**

- Instead of **methods**, Perl uses **subroutines**

Subroutines

Subroutines are defined as follows in Perl:

```
sub identifier {  
    statements  
}
```

Assignment Project Exam Help

- Subroutine identifier should be All sub
 - The return statement can be used to terminate the execution of a subroutine to make value the return value of the subroutine
 - If the execution of a subroutine terminates without encountering a return statement, then the value of the last evaluation of an expression in the subroutine is returned

The return value does not have to be scalar value, but can be a list

Parameters and Arguments

Subroutines are defined as follows in Perl:

```
sub identifier {  
    statements  
}
```

Assignment Project Exam Help

- In Perl t
(or thei
~ th <https://eduassistpro.github.io>
- Arguments are passed to a subroutine via a special ar
- Individual arguments are accessed using
- It is up to the subroutine to process arguments as is app
- The array @_ is private to the subroutine
~ each nested subroutine call gets its own @_ array

Add WeChat edu_assist_pro

Parameters and Arguments: Examples

- The Java method

```
public static int sum2( int f, int s) {  
    f = f+s;  
    return f;  
}
```

could be

```
sub sum2 {  
    ret  
}
```

<https://eduassistpro.github.io>

- A more general solution, taking into account that a given arbitrarily many arguments, is the following:

```
1 sub sum {  
2     return undef if (@_ < 1);  
3     $sum = shift(@_);  
4     foreach (@_) { $sum += $_ }  
5     return $sum;  
6 }
```

Private variables

```
sub sum {  
    return undef if (@_ < 1);  
    $sum = shift(@_);  
    foreach (@_), { $sum += $_ }  
    return $sum;  
}
```

Assignment Project Exam Help

The varia

```
$sum = 0;  
print "Value  
print "Return value of sum: ",&sum(5,4,3,2,1), "\n";  
print "Value of \$sum after call of sum: ",$sum, "\n"
```

produces the output

Value of \$sum before call of sum: 5

Return value of sum: 15

Value of \$sum after call of sum: 15

Add WeChat edu_assist_pro

This use of **global** variables in subroutines is often undesirable

~ we want \$sum to be **private/local** to the subroutine

Private variables

- The operator `my` declares a variable or list of variables to be `private`:

```
my $variable;  
my ($variable1, $variable2);  
my @array;
```

Assignment Project Exam Help

- Such a declaration can be combined with a (list) assignment:

```
my $v  
my ($  
my @a
```

<https://eduassistpro.github.io>

- Each `call` of a subroutine will get its own `copy` of its `private`

Example: Add WeChat `edu_assist_pr`

```
sub sum {  
    return undef if (@_ < 1);  
    my $sum = shift(@_);  
    foreach (@_) { $sum += $_ }  
    return $sum;  
}
```

Calling a subroutine

A subroutine is **called** by using the subroutine name with an ampersand **&** in front possibly followed by a list of arguments

The ampersand is optional if a list of arguments is present

```
sub identifier {  
    statements  
}
```

```
... & id  
... & id  
... identifier(arguments) ...
```

Add WeChat edu_assist_pr

```
print "sum0: ",&sum ,"\n";  
print "sum0: ",sum(), "\n";  
print "sum1: ",sum(5), "\n";  
print "sum2: ",sum(5,4), "\n";  
print "sum5: ",&sum(5,4,3,2,1), "\n";  
$total = sum(9,8,7,6)+sum(5,4,3,2,1);  
sum(1,2,3,4);
```

Persistent variables

- Private variables within a subroutine are forgotten once a call of the subroutine is completed
- In Perl 5.10 and later versions, we can make a variable both private and persistent using the state operator

~ th

in

<https://eduassistpro.github.io>

Example

```
use 5.010;  
sub running_sum {  
    state $sum;  
    foreach (@_) { $sum += $_ }  
    return $sum;  
}
```

Add WeChat edu_assist_pro

Persistent variables

Example:

```
1 use 5.010;
2
3 sub running_sum {
4     state $sum;
5     foreach (@_) {
6         $sum += $_;
7     }
8     return $sum;
9 }
10
11 print "running_sum():\t\t",    running_sum(),      "\n";
12 print "running_sum(5):\t\t",    running_sum(5),     "\n";
13 print "running_sum(5,4):\t\t",   running_sum(5,4),   "\n";
14 print "running_sum(3,2,1):\t\t", running_sum(3,2,1), "\n";
```

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

Output:

```
running_sum():
running_sum(5):          5
running_sum(5,4):        14
running_sum(3,2,1):      20
```

Nested subroutine definitions

- Perl allows nested subroutine definitions (unlike C or Java)

```
sub outer_sub {  
    sub inner_sub { ... }  
}
```

Assignment Project Exam Help

- Normal
~> the inner subroutine can be called from outside its definition.
- However, Perl allows inner subroutines to be called from outside their package (within the package in which they are defined).

```
Add WeChat edu_assist_pro
```

```
sub outer_sub {  
    sub inner_sub { ... }  
}  
inner_sub();
```

Nested subroutine definitions

If an **inner subroutine** uses a **local variable** of an **outer subroutine**, then it refers to the **instance** of that local variable created the first time the outer subroutine was called.

Assignment Project Exam Help

```
sub outer {  
    my $x = $_[0];  
    sub inner {  
        return $x;  
    }  
    print "1: ", outer(10), "\n";  
    print "2: ", outer(20), "\n";  
}
```

```
1: 10  
2: 10 # not 20!
```

- ~ Do **not** refer to local variables of an outer subroutine, pass information via arguments instead

Nested subroutine definitions: Example

```
sub sqrt2 {  
    my $x = shift(@_);  
    my $precision = 0.001;  
    sub sqrtIter {  
        my ($guess, $x) = @_;  
        if (isGoodEnough($guess, $x)) {  
            return $guess;  
        } else {  
            my $newGuess = improveGuess($guess, $x);  
            return sqrtIter($newGuess, $x);  
        }  
    }  
    return sqrtIter(1.0, $x);  
}
```

Assignment Project Exam Help
<https://eduassistpro.github.io/>
Add WeChat edu_assist_pro

Revision

Read

Assignment Project Exam Help

- Chapter 9 Processing Text with Regular Expressions

- Cha

of

<https://eduassistpro.github.io/>

R. L. Sch

Learning Perl.

O'Reilly, 2011

Add WeChat edu_assist_pro

- <http://perldoc.perl.org/perlsub.html>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 7: Perl (Part 6)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

Assignment Project Exam Help

13 Input/Output

File handles

Open

Clo

Re

Sel

Pri

Here documents

14 Arguments and Options

Invocation Arguments

Options

Add WeChat edu_assist_pr

I/O Connections

- Perl programs interact with their environment via I/O connections
- A filehandle is the name in a Perl program for such an I/O connection, given by a Perl identifier
- There are

STD	https://eduassistpro.github.io/
STDERR	Standard Error, for error output, typically defaults to the terminal
DATA	Input from data stored after Perl program
ARGV	Iterates over command-line filenames in @ARGV
ARGVOUT	Points to the currently open output file when doing edit-in-place processing with -i <code>perl -pi -e 's/cat/dog/' file</code>

I/O Connections

Except for the six predefined I/O connections, all other I/O connections

- need to be `opened` before they can be used

`open <filehandle>, mode expr`

- should be `closed` once no longer needed

`clos`

- can be used to

`<filehandle>`

- can be used to `write to`

`print filehandle list`

`printf filehandle list`

- can be selected as default output

`select filehandle`

I/O Connections

Example:

```
open INPUT, "<", "oldtext.txt" or die "Cannot open file";
open OUTPUT, ">", "newtext.txt";
while (<INPUT>) {
    s!/(\d+) degrees Fahrenheit!
    $p
    print 0
}
close(INPUT);
close(OUTPUT);
```

oldtext.txt:

105 degrees Fahrenheit is quite warm

newtext.txt:

41 degrees Celcius is quite warm

Opening a filehandle

`open filehandle, expr`

`open filehandle, mode, expr`

• Opens an I/O connection specified by *mode* and *expr* and associates it with *filehandle*

- *expr*
- *mode*

Mod			
<	read file		
>	write file	yes	
>>	append file	yes	
+<	read/write file		
+>	read/write file	yes	yes
+>>	read/append file	yes	
-	write to command	yes	
- !	read from command	yes	

Closing a filehandle

`close`

`close filehandle`

- Flushes the I/O buffer and closes the I/O connection associated with *filehandle*

- Return value

- Close

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Reading

<filehandle>

- In a scalar context, returns a string consisting of all characters from the filehandle up to the next occurrence of \$/ (the input record separator)
- In a list context, returns a list of filehandle (Default value is STDIN)

of filehandle
(Default value is STDIN)

```
1 open INPUT, "<", "oldtext.txt" or die "Cannot open file";
2 $first_line = <INPUT>;
3 while (<another_line = <INPUT>>) { ... }
4 close INPUT;
5
6 open LS, "-|", "ls -1";
7 @files = <LS>;
8 close LS;
9 foreach $file (@files) { ... }
```

Add WeChat edu_assist_pro

Selecting a filehandle as default output

`select`

`select filehandle`

- If *filehandle* is supplied, sets the new current default filehandle for output

 ~`wr`

 ~`Re`

- Returns

dle

ehandle

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Printing

```
print filehandle list
print filehandle
print list
print
```

Assignment Project Exam Help

- Print a s
- If *fil*
- If *lis*
- The current value of \$, (if any) is printed bet
(Default: `undef`)
- The current value of \$ \ (if any) is printed aft
been printed
(Default: `undef`)

Add WeChat edu_assist_pr

Printing: Formatting

`sprintf(format, list)`

- Returns a string formatted by the usual printf conventions of the C library function `printf` (but does not by itself print anything)

```
sprintf " (%.10.3f)" 1234.5678
```

forma

and put

```
( 1234.568)
```

Add WeChat `edu_assist_pr`
See <http://perldoc.perl.org/functions/sprintf.html> for further details

Printing: Formatting

`printf filehandle format, list`

`printf format, list`

• Equivalent to

`print filehandle sprintf(format, list)`

except

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Printing: Formatting

Format strings can be stored in variables and can be constructed on-the-fly:

```
@list = qw(wilma dino pebbles);
$format = "The items are:\n%10s\n" x @list;
printf $format, @list;
```

Output:

```
The items are:
    wi
    dino
    pebbles
```

(The code above uses the `quote word` function to generate a list of words.

See http://perlmeme.org/howtos/perlfunc/qw_function.html for details)

Here documents

- A **here document** is a way of specifying multi-line strings in a scripting or programming language
- The basic syntax is

```
<<identifier  
here docu  
identi  
ide
```

- **https://eduassistpro.github.io**
- **here document ends**
- **identifier** might optionally be surrounded by quotes or backticks
An unquoted identifier works like a double-quoted string
- The **here document** starts on the following line
- The **terminating string identifier** must appear by itself (unquoted and with no surrounding whitespace) after the last line of the **here document**

Here documents: Double-quotes

```
$title = "My HTML document"
print <<"END";
Content-type: text/html
!DOCTYPE html
<HTML>
<HEADER><TITLE>$title</TITLE></HEADER>
<BODY>
  <H1>$tit
  Lots o HTML mac
</BODY>
</HTML>
END
```

```
Content-type: text/html
!DOCTYPE html
<HTML>
<HEADER><TITLE>My HTML document</TITLE></HEADER>
<BODY>
  <H1>My HTML document</H1>
  Lots of HTML markup here
</BODY>
</HTML>
```

The double-quotes in "END" indicate that everything between the opening "END" and closing "END" should be

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Here documents: Single-quotes

```
$title = "My\u2022HTML\u2022document"
print <<'END';
Content-type: text/html
<!DOCTYPE html>
<HTML><HE
<BODY></B
END
```

The singl

END should be treated like a single-quoted string

~ no variable interpolation is applied

~ \$title will not be expanded

```
Content-type: text/html
```

```
<!DOCTYPE html>
<HTML><HEADER><TITLE>$title</TITLE></HEADER>
<BODY></BODY></HTML>
END
```

Here documents: Backticks

```
$command = "ls";
print <<'END';
$command`1
END
```

Assignment Project Exam Help

The **backticks** in ‘END’ tell Perl to run the **here document** as a **shell script** (with the h

```
handouts
handouts.
handouts.pdf
handouts.tex
```

Add WeChat edu_assist_p

Here documents: Variables

Here documents can be assigned to variables and manipulated using string operations

```
$header = <<"HEADER";  
Content-type: text/html
```

```
<!DOCTYPE ht  
<HTML><HE  
HEADER  
$body = <<"BODY";  
<BODY>  
    <H1>$title</H1>  
    Lots of HTML markup here  
</BODY>  
</HTML>  
BODY  
  
$html = $header.$body;  
print $html;
```

Add WeChat edu_assist_pro

Invocation Arguments

- Another way to provide input to a Perl program are invocation arguments (command-line arguments)

Assignment Project Exam Help

- The invocation arguments given to a Perl program are stored in the special variable `ARGV`.

```
perl https://eduassistpro.github.io/Assignment/Perl/args.pl  
print "Number of arguments: " . scalar(@ARGV);  
for ($index=0; $index <= $#ARGV; $index++) {  
    print "Argument $index: $ARGV[$index],\n";  
}  
./perl_program1 ada 'bob' 2
```

Add WeChat edu_assist_pro

Output:

```
Number of arguments: 3  
Argument 0: ada  
Argument 1: bob  
Argument 2: 2
```

Options

- There are various Perl modules that make it easier to process command-line options

`-s scale=5 -debug -file='image.png'`

- One such module is `Getopt::Long`:

http

- The m

- GetO

@ARGV according to an option specification

- Arguments that do not fit to the option specification

ARGV

- GetOptions returns true if @ARGV can be

Add WeChat edu_assist_pr

Options: Example

perl_program2:

```
use Getopt::Long;
my $file = "photo.png";
my $scale = 2;
my $debug = 0;

$result = GetOpt::Long::GetOptions("file=s", "scale=i", "debug");
print "Debug: $debug; Scale: $scale; File: $file\n";
print "Number of arguments: $#ARGV+1\n";
print "Arguments: ", join(' ', @ARGV), "\n";

./perl_program2 --scale=5 --file='image.png' arg1 arg2
```

```
Debug: 0; Scale: 5; File: image.png
Number of arguments: 2
Arguments: arg1, arg2
```

Revision

Read

Assignment Project Exam Help

of

R. L. Sch

Learni

O'Reilly, 2011.

Add WeChat edu_assist_pr

- <http://perldoc.perl.org/perlop.html>
- <http://perldoc.perl.org/perlop.html#Quote-Like-Operators>
- <http://perldoc.perl.org/Getopt/Long.html>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 8: Perl (Part 7)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

Assignment Project Exam Help

15 CGI

Overview

CG I

<https://eduassistpro.github.io>

16 The Perl module CGI.pm

Motivation

HTML shortcuts

Forms

Add WeChat edu_assist_pm

Common Gateway Interface — CGI

The **Common Gateway Interface** (CGI) is a standard method for web servers to use an external application, a **CGI program**, to **dynamically generate web pages**.

- ① A **Web client** generates a **client request**, for example, from a **HTML form**, and sends it to a **web server**

- ② The **w**
conv
- ③ The **C**

the server passes the **program's response** back to **t**

Add WeChat edu_assist_pr

Client requests

In the following we focus on **client requests** that are generated using **HTML forms**

Assignment Project Exam Help

```
<!DOCTYPE html>
<html>
<head><title>My HTML Form</title></head>
<body>
<form action
  "http://"
  method=""

<label>Enter your user name:
  <input type="text" name="username"></label><br>
<label>Enter your full name
  <input type="text" name="fullname"></label><br>
<input type="submit" value="Click for response">
</form>
</body>
</html>
```

Add WeChat edu_assist_p

<https://eduassistpro.github.io/>

Client requests

In the following we focus on **client requests** that are generated using **HTML forms**

```
<!DOCTYPE html>
<html>
<head><title>My HTML Form</title></head>
<body>
<form action="h
method="pos
<label>Enter yo
<label>Enter yo
<input type="su
</form>
</body>
</html>
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_p

Encoding of input data

- Input data from an HTML form is sent URL-encoded as sequence of key-value pairs: key1=value1&key2=value2&...

Example:

username=dave&fullname=David%20Davidson

- All characters are encoded using ASCII codes (preceded by %)
- ASCII characters that are not unreserved characters are encoded using ASCII codes (preceded by %)
 - A space is represented as %20 or +
 - + is represented as %2B
 - % is represented as %25

Examples:

username=cath&fullname=Catherine+0%27Donnell

Request methods: GET versus POST

The two main request methods used with HTML forms are **GET** and **POST**:

Assignment Project Exam Help

- **GET:**

<

<

https://eduassistpro.github.io

- **For**

Example:

```
GET /cgi-bin/cgiwrap/tlrich/demo?username=davex  
fullname=David+Davidson HTTP/1.1  
Host: cgi.csc.liv.ac.uk
```

Request methods: GET versus POST

The two main request methods used with HTML forms
are **GET** and **POST**:

Assignment Project Exam Help

- **POST**
- **For**
- **For**

https://eduassistpro.github.io

Example

```
POST /cgi-bin/cgiwrap/ullrich/demo HTTP/1.1
```

```
Host: cgi.csc.liv.ac.uk
```

Add WeChat edu_assist_pr

```
username=dave&fullname=David+Davidson
```

Environment variables: GET

Env variable	Meaning
QUERY_STRING	The query information passed to the program
REQUEST_METHOD	The request method that was used
PATH_INFO	Extra path information passed to a CGI program
PATH_	Physical
SCRIPT_NAME	
SCRIPT_FILENAME	

Example (1):

```
GET http://loci.cs.wisc.edu/cgi-bin/cgiwrap/ullrich/demo/more/dirs?  
username=dave&fullname=David+Davidson
```

```
QUERY_STRING          username=dave&fullname=David+Davidson  
REQUEST_METHOD        GET  
PATH_INFO             /more/dirs  
PATH_TRANSLATED       /users/www/external/docs/more/dirs  
SCRIPT_NAME           /cgi-bin/cgiwrap/ullrich/demo  
SCRIPT_FILENAME       /users/loco/ullrich/public_html/cgi-bin/demo
```

STDIN

```
# empty
```

Environment variables: GET

Env variable	Meaning
QUERY_STRING	The query information passed to the program
REQUEST_METHOD	The request method that was used
PATH_INFO	Extra path information passed to a CGI program
PATH_	Physical
SCRIPT_NAME	
SCRIPT_FILENAME	

Example (2):

```
GET http://loci.cs.wisc.edu/cgi-bin/cgiwrap/ullrich/demo/more/dirs?  
username=2%60n+d%2Bt+e+s%27t&fullname=Peter+Newton
```

```
QUERY_STRING          username=2%60n+d%2Bt+e+s%27t&fullname=Peter+Newton  
REQUEST_METHOD        GET  
PATH_INFO             /more/dirs  
PATH_TRANSLATED       /users/www/external/docs/more/dirs  
SCRIPT_NAME           /cgi-bin/cgiwrap/ullrich/demo  
SCRIPT_FILENAME        /users/loco/ullrich/public_html/cgi-bin/demo
```

STDIN

```
# empty
```

Environment variables: POST

Env variable	Meaning
QUERY_STRING	The query information passed to the program
REQUEST_METHOD	The request method that was used
SCRIPT_NAME	The relative virtual path of the CGI program
SCRIP	

Example

```
POST /cgi-bin/cg
Host: cgi.csc.liv.ac.uk
```

```
username=2%60n+d%2Bt+e+s%27t&fullname=Peter+Newton
```

```
QUERY_STRING
# empty
REQUEST_METHOD POST
```

```
SCRIPT_NAME /cgi-bin/cgiwrap/ullrich/demo
SCRIPT_FILENAME /users/loco/ullrich/public_html/cgi-bin/demo
```

```
STDIN username=2%60n+d%2Bt+e+s%27t&fullname=Peter+Newton
```

Add WeChat edu_assist_p

More environment variables

Env variable	Meaning
HTTP_ACCEPT	A list of the MIME types that the client can accept
HTTP_REFERER	The URL of the document that the client points to before accessing the CGI program
HTTP_	
REMOTE_HOST	The remote hostname o
SERVER_NAME	The server's host name
SERVER_PORT	The port number of the ho is running
SERVER_SOFTWARE	The name and version of the server software

CGI programs and Perl

- CGI programs need to process input data from environment variables and STDIN, depending on the request method
 - preferably the input data would be accessible by the program in a uniform way

Assignment Project Exam Help

- CGI pr

~> pr

- CGI pr
 - preferably, there would be an easy way to produce

Add WeChat edu_assist_pr

In Perl all this can be achieved with the use of the

<http://perldoc.perl.org/CGI.html>

CGI.pm HTML shortcuts

- CGI.pm provides so-called **HTML shortcuts** that create **HTML tags**

a	address	applet	b	body	br	center	code
dd	div	dl	dt	em	font	form	hr
h1	h2	h3	h4	h5	h6	head	header
html							strong
sup							ul

<https://eduassistpro.github.io/>

- **HTML tags** have **attributes** and **contents**

```
<p align="right">This is a paragraph</p>
```

Add WeChat edu_assist_pm

- **HTML shortcuts** are given

- **HTML attributes** in the form of a **hash reference** as the first argument
- the **contents** as any subsequent arguments

```
p({-align=>right}, "This is a paragraph")
```

CGI.pm HTML shortcuts: Examples

Code: `print p();`

Output: `<p />`

Code: `print p('');`

Output: `<p></p>`

Code: `print p({-align`

Output: `<p align`

Code: `print p({-class=>right_para,-id=>p1}, "Text");`

Output: `<p class='right_para' id='p1'>Text</p>`

Add WeChat edu_assist_pro

CGI.pm HTML shortcuts: Nesting vs Start/End

- Nested HTML tags using nested HTML shortcuts

```
Code:    print p(em("Emphasised") . "Text"), "\n";
```

```
Output: <p><em>Emphasised</em> Text</p>
```

Assignment Project Exam Help

- Neste

```
use CGI qw(-ut);
print start_
    "Text", end_p(), "\n";
```

```
Output: <p><em>Emphasised</em> Text</p>
```

Add WeChat edu_assist_pro

The following `start_tag`/`end_tag` HTML shortcuts are generated automatically by CGI.pm:

```
start_html(), start_form(), start_multipart_form()
end_html(),   end_form()      end_multipart_form()
```

All others need to be requested by adding `*tag` to the CGI.pm import list

CGI.pm Forms

- HTML forms are created using start_form and end_form

```
print start_form({-method=>request_method,  
                  -action=>uri});  
[form elements]  
print end_form;
```

Assignment Project Exam Help

- HTM

tex
fil
popup_menu
image_button
radio_group

https://eduassistpro.github.io

text	input
file	file
popup_menu	optgroup
image_button	checkbox
radio_group	reset

- optgroup creates an option group within a popup menu
 - optgroup occurs nested inside popup_menu
- All other HTML shortcuts for HTML form elements will occur independently of each other within a form

CGI.pm Forms: Examples

```
printtextfield({-name=>'username',
                 -value=>'dave',
                 -size=100,
                 -maxlength=500});
```

Assignment Project Exam Help

- `-nam` and is the name of the field
- `-val` is the value of the field
- `-size` is the size of the text field in characters
- `-maxlength` is the maximum number of characters the field will accept

Output:

```
<input type="text" name="username"
       value="dave" size="100" maxlength="500" />
```

CGI.pm Forms: Examples

```
print submit({-name=>'submit',  
             -label=>'Click for response'});
```

- `-name` is an optional argument that allows to distinguish submit buttons from each other

- `-lab`

show

<https://eduassistpro.github.io/>

Output:

```
<input type='submit' name='submit'  
       value='Click for response' />
```

CGI.pm Forms: Example

```
#!/usr/bin/perl

use CGI qw(-utf8 :all);

print header(-charset=> utf-8),
        start_html({-title=>'My HTML Form',
                    -author=>'u.hustadt@liverpool.ac.uk',

print start_form;

print textfield({-name=>'username',
                -value=>'dave',
                -size=>100});

print br();
print textfield({-name=>'fullname',
                -value=>'Please enter your name',
                -size=>100});

print br();
print submit({-name=>'submit',
              -value=>'Click for response'});
print end_form, end_html;
```

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Making it work

For CGI programs to work on our systems you must proceed as follows:

- ① Your home directory must be 'world executable'
- ② You must have a directory

`$HOME/public_html/cgi-bin/`

- You can then `exec` your script.
- ③ You can then access your script via a URL:

`$HOME/public_html/cgi-bin/`

- and must be `executable` by everyone.
- ④ The CGI Script can then be accessed using the URLs:
`http://csci.csc.liv.ac.uk/cgi-bin/cgiwrap/<user>/<script>`
or `http://csci.csc.liv.ac.uk/cgi-bin/cgiwrapd/<user>/<script>`

where `<user>` is your user name

and `<script>` is the filename of the script

(`cgicgiwrapd` provides debugging output, but does not reveal all errors)

Accessing and processing data

- Perl provides a hash `%ENV` that stores the information stored in environment variables
- Processing `%ENV` is done in the standard way for hashes

Assignment Project Exam Help

```
print "The request method used is ",
```

```
foreach $key ()  
    print "Th  
}  
}
```

<https://eduassistpro.github.io/>

Output:

The request method used is GET
The value of SCRIPT_NAME is /cgi-bin/cgiwrap/ullrich/demo
The value of SERVER_NAME is cgi.csc.liv.ac.uk
The value of SERVER_ADMIN is root@localhost
The value of HTTP_ACCEPT_ENCODING is gzip,deflate
The value of HTTP_CONNECTION is keep-alive
The value of REQUEST_METHOD is GET

Accessing and processing data

- CGI.pm provides the `param` routine to access the input data of HTML forms
- For a sequence of key-value pairs

`key1=value1&key2=value2&key3=value3&...`

repres

<https://eduassistpro.github.io/>

will ret

`value1`

`value2`

`value3`

while `param()` returns the list '`key1`'

- The `values` returned by `param` have already been converted to strings
- `param('key')` returns the empty string if `value` is empty
- `param('key')` returns `undef` if `key` is not among the key-value pairs of the request
- This does not depend on whether the `request method` is GET or POST

Accessing and processing data

- `CGI.pm` provides the `param` routine to access the input data of HTML forms

Assignment Project Exam Help

```
print "The value of username is ",  
      param('username'), br(), br(), "\n";  
print "The v  
  
foreach $key {  
    print "The value of $key is ", param($key), br(), "\n";  
}  
}
```

Output: Add WeChat edu_assist_pr

```
The value of username is dave  
The value of fullname is David Davidson
```

```
The value of submit is Click for response  
The value of username is dave  
The value of fullname is David Davidson
```

Accessing and processing data: UTF-8

- The `pragma -utf8` in

```
use CGI qw(-utf8 :all);
```

makes makes CGI.pm treat all param() values as UTF-8 strings

- Alternatively, specific param() values can be decoded using the decode subroutines

```
use Encode;
my $fulln
```

<https://eduassistpro.github.io/>

- With

```
binmode(STDOUT, ":encoding(utf-8)");
print header(-charset => 'utf-8');
```

Add WeChat edu_assist_p

we ensure that the web page we produce is sent to the browser using UTF-8 encoding

Accessing and processing data: Security

- Do **not** trust any data accessed via param (beware **code injection**)

Example:

```
print "The value of username is ", param('username'), "\n";  
together with input
```

Assignment Project Exam Help

```
<script>window.location="http://malware_site/"</script>
```

for us

- Check

<https://eduassistpro.github.io>

```
if (param ~  
    print "Not a valid user name"  
) else {  
    print "The value of username is ", param('username')  
}
```

Add WeChat edu_assist_pro

or sanitise the input using the CGI.pm routine escapeHTML:

```
print "The value of username is ",  
      escapeHTML(param('username')), "\n";
```

or even better, do both

CGI.pm Scripts: Example (Part 1)

```
use CGI qw(-utf-8 :all *table);
binmode(STDOUT, ":encoding(utf-8)");

print header(-charset=>'utf-8'), "\n",
      start_html({-title=>'Form Processing',
                  -author=>'u.hustadt@liverpool.ac.uk'});

if (!defined($ENV{'HTTP_REFERER'})) {
    # This branch is executed if the client request is generated
    # by the form
    print start_html,
          print textfield({-name=>'fullname',
                           -value=>'Please enter your name',
                           -size=>100}), "\n";
    print br(), "\n";
    print submit({-name=>'submit',
                  -value=>'Click for response'}), "\n";
    print end_form;
} else {
    # This branch is executed if the client request is generated
    # by the form
}
```

Assignment Project Exam Help
<https://eduassistpro.github.io>

Add WeChat edu_assist_pm

CGI.pm Scripts: Example (Part 2)

```
# (We are in the else-branch now)

print start_table({-border=>1});
print caption("Inputs");
foreach $key (param()) {
    print Tr(td('PARAM'),td($key),td(escapeHTML(param($key))));
}
foreach $ke
{
    print Tr
}
print end_t
}
print end_html;
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

CGI.pm Scripts: Example (Part 3)

Page produced on the first visit

Assignment Project Exam Help

Page produced

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Revision

Read
Assignment Project Exam Help

- Chapter 11: Perl Modules

of

R. L. Scott
Learning Perl.
O'Reilly, 2011

Add WeChat edu_assist_pm

- <http://perldoc.perl.org/CGI.html>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 9: PHP (Part 1)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

17 PHP

Motivation

Assignment Project Exam Help

18 Overview

Features

API

<https://eduassistpro.github.io>

19 Types and Variables

Type

Variables

Add WeChat edu_assist_pro

Type juggling and Type casting

Comparisons

Common Gateway Interface — CGI

The **Common Gateway Interface** (CGI) is a standard method for web servers to use external applications, a **CGI program**, to dynamically generate web pages

Assignment Project Exam Help

- ① A **Web client** generates a **client request**, for example, from a **HTML form**, and sends it to a **web server**
- ② The **w**
conv
- ③ The **C**
the **program's response** back to the client

Add WeChat edu_assist_pr

Disadvantages of CGI/Perl

- A distinction is made between static web pages and dynamic web pages created by an external program
 - Using Perl scripting it is difficult to add 'a little bit' of dynamic content to a web page
- Use of a
 - star
 - exchanging data between web server and external p

~ resource-intensive

Add WeChat edu_assist_pro

If our main interest is the creation of dynamic web pages, then the scripting language we use

- should integrate well with HTML
- should not require a web server to execute an external program

PHP

- PHP is (now) a recursive acronym for PHP: Hypertext Preprocessor
- Development started in 1994 by Rasmus Lerdorf
- Originally designed as a tool for tracking visitors at Lerdorf's website
- Devel
- server
- Inheri
- Easy-to-use interface to databases
- Free, open-source
- Probably the most widely used server-side web pro
- Negatives: Inconsistent, muddled API; no scalar objects

Add WeChat edu_assist_pro

The departmental web server uses PHP 5.6.25 (released August 2014)
PHP 7 was released in December 2015 (PHP 6 was never released)

PHP processing

- Server plug-ins exist for various web servers
 - ~ avoids the need to execute an external program
- PHP code is embedded into HTML pages using tags
 - ~ static web pages can easily be turned into dynamic ones

PHP satisfies

Process

<https://eduassistpro.github.io/>

- ① The web server receives a client request
- ② The web server recognizes that the client request is a HTML page containing PHP code
- ③ The server executes the PHP code, substitutes output into the HTML page, the resulting page is then sent to the client

As in the case of Perl, the client never sees the PHP code, only the HTML web page that is produced

PHP: Applications

- Applications written using PHP
 - activeCollab – Project Collaboration Software
<http://www.activecollab.com/>
 - Drupal – Content Management System (CMS)
<http://drupal.org/home>
 - Mediawiki – eCommunity Platform
http://www.mediawiki.org/wiki/Main_Page
 - Moodle – Virtual Learning Environment (VLE)
<http://moodle.org>
 - SugarCRM – Customer Relationship Management (CRM) platform
<http://www.sugarcrm.com/crm/>
 - WordPress – Blogging tool and CMS
<http://wordpress.org/>

Assignment Project Exam Help

Add WeChat edu_assist_pro

PHP: Websites

- Websites using PHP:
 - Delicious – social bookmarking
<http://delicious.com/>
 - Digg – social news website
<http://digg.com>
 - FaceBook – social networking
<http://www.facebook.com>
 - Flickr – photo sharing
<http://www.flickr.com>
 - Friendster – social gaming
<http://www.friendster.com>
 - SourceForge – web-based source code repository
<http://sourceforge.net/>
 - Wikipedia – collaboratively built encyclopedia
<http://www.wikipedia.org>

Assignment Project Exam Help

Add WeChat edu_assist_pr

Recommended texts

- R. Nixon:

Learning PHP, MySQL, and JavaScript,
O'Reilly, 2009.

Har

(or lat

<https://eduassistpro.github.io/>

- M. Achour, F. Betz, A. Dovgal, N. Lopes,
H. Magnusson, G. Richter, D. Seguy, J.
[PHP Manual](#).

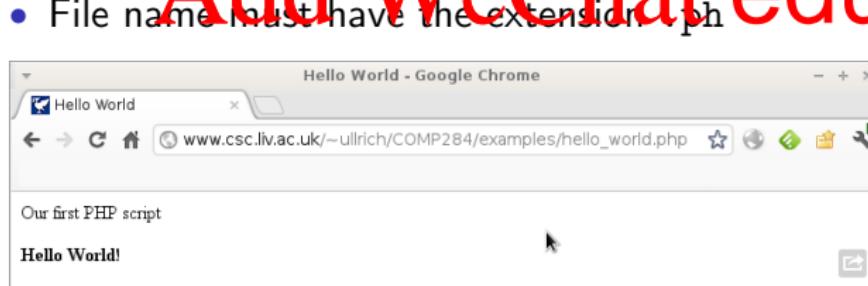
PHP Documentation Group, 2018.

<http://www.php.net/manual/en/index.php>

PHP: Hello World!

```
1 <html>
2 <head><title>Hello World</title></head>
3 <body>
4 <p>Our first PHP script!</p>
5 <?php
6   print ("<p><b>Hello World!</b></p>\n");
7 ?>
8 </body>
```

- PHP code
- File must be stored in a directory accessible by the web example \$HOME/public_html and be readable
- File name must have the extension .php



PHP: Hello World!

Since version 4.3.0, PHP also has a command line interface

```
1 #!/usr/bin/php
2 <?php
3 /* Author: Ulrich Hustadt
4      A "Hello World" PHP script. */
5 prin
6 // A sing
7 ?>
```

<https://eduassistpro.github.io/>

- PHP code still needs to be enclosed between `<?php` and `?>`
 - Code must be stored in an executable file
 - File name does not need to have any particular format
- ~ PHP can be used as **scripting language** outside a web programming context

Output:

Hello World!

PHP: Hello World!

```
<html>
<head><title>Hello World</title></head>
<body><p>Our first PHP script</p>
<?pho
print '<p><b>Hello World!</b></p>'<n">',
?>
</body></
```

- Can also run:

`php filename`

- File does not need to be executable, only readable for the server.

Output:

Add WeChat edu_assist_pro

```
<html>
<head><title>Hello World</title></head>
<body><p>Our first PHP script</p>
<p><b>Hello World!</b></p>
</body></html>
```

PHP scripts

- PHP scripts are typically embedded into HTML documents and are enclosed between `<?php` and `?>` tags
- A PHP script consists of one or more statements and comments
~ there is no need for a main function (or classes)

- Stat

- Whi
(Th

<https://eduassistpro.github.io>

?>

- One

// #

- Multi-line comments are enclosed in /* a

Add WeChat edu_assist_pr

Types

PHP has eight primitive types

- Four scalar types:

bool – Booleans

- int – integers
- flo
- str

- Two compound types:

array – arrays

object – objects

NULL

<https://eduassistpro.github.io>

- Integers, floating-point numbers, and strings do not inherit from the corresponding Perl scalars, including the single-quoted versus double-quoted strings
- In contrast to Perl, PHP does distinguish between different types including between the four scalar types

Variables

- All PHP variable names start with \$ followed by a PHP identifier
- A PHP identifier consists of letters, digits, and underscores, but cannot start with a digit
PHP identifiers are case sensitive

- In PHP

- A variable

although initialisation is a good idea

- Uninitialized variables have a default value of their type in the context in which they are used

Add WeChat edu_assist_pro

Type	Default	Type	Default
<u>bool</u>	FALSE	<u>string</u>	empty string
<u>int / float</u>	0	<u>array</u>	empty array

If there is no context, then the default value is NULL

Assignments

- Just like Java and Perl, PHP uses the equality sign = for assignments

```
$student_id = 200846369;
```

As in Perl, this is an assignment expression

- The value of an assignment expression is the value assigned

```
$b = ($a = 0) + 1;  
// $a  
// $b
```

Add WeChat edu_assist_pro

Binary assignments

PHP also supports the standard **binary assignment** operators:

Binary assignment	Equivalent assignment
$\$a += \b	$\$a = \$a + \$b$
$\$a -= \b	$\$a = \$a - \$b$
$\$a **= \b	$\$a = \$a ** \$b$
$\$a .= \b	$\$a = \$a . \$b$

Example:

```
// Convert Fahrenheit to Celsius:  
// Subtract 32, then multiply by 5, then divide by 9  
$temperature = 105; // temperature in Fahrenheit  
$temperature -= 32;  
$temperature *= 5/9; // converted to Celsius
```

Constants

- `bool define(string, expr [, case_insensitive])`
 - defines a constant that is globally accessible within a script
String should be a string consisting of a PHP identifier (preferably all upper-case)
 - The PHP identifier is the `name` of the constant
 - `exp`
 - `cas`
 - `whe`
 - returns TRUE on success or FALSE on failure

```
define("PI", 3.14159);  
define("SPEED_OF_LIGHT", 299792458, true);
```

Add WeChat edu_assist_pro

Constants

- To use a constant we simply use its **name**

```
define("PI", 3.14159);
define("SPEED_OF_LIGHT", 299792458, true);
$circumference = PI * $diameter;
$distance      = speed_of_light * $time;
```

- Cavea

<https://eduassistpro.github.io/>

```
print "1 - Value of PI: PI\n";
print "2 - Value of PI: ".PI."\n";
```

```
1 - Value of PI: PI
2 - Value of PI: 3.14159
```

Add WeChat edu_assist_pro

Values, Variables and Types

PHP provides several functions that explore the type of an expression:

string gettype(expr)

bool is_type(expr)

void var_dump(expr)

returns the type of expr as string

checks whether expr is of type type

displays structured information about expr

```
<?php print "T  
print "Type of \"23\": ".gettype("23")."\n";  
if (is_int(23)) { echo "23 is an integer\n"; }  
else { echo "23 is not an integer\n"; }  
?>
```

```
Type of 23: integer  
Type of 23.0: double  
Type of "23": string  
23 is an integer
```

Type juggling and Type casting

- PHP automatically converts a value to the appropriate type as required by the operation applied to the value (type juggling)

Assignment Project Exam Help

```
3 "Worlds" 6 "2Worlds"  
"2" * 3 6
```

```
"1.2
```

```
"hel
```

```
"10h
```

<https://eduassistpro.github.io>

- PHP also supports explicit type casting via

(int) 12"	12	(FALSE
(int) "1.23e2"	1	(TRUE
(int) ("1.23e2" + 0)	123	(float) "1.23e2"	~> 123
(int) "10hello5"	10		
(int) 10.5	10		
(array) "foo"		array(0 => "foo")	

Comparison operators

Type juggling also plays a role in the way PHP comparison operators work:

<code>expr1 == expr2</code>	Equal	TRUE iff <code>expr1</code> is equal to <code>expr2</code> after type juggling
<code>expr1 != expr2</code>	Not equal	TRUE iff <code>expr1</code> is not equal to <code>expr2</code> after type juggling
<code>expr1 <> expr2</code>	Not equal	TRUE iff <code>expr1</code> is not equal to <code>expr2</code>
<code>expr1</code> <code>expr1</code>		https://eduassistpro.github.io ,

Note: For `==`, `=`, and `<>`, numerical strings are

and compared numerically

"123" == 123	~>	TRUE	"123" === 123	~>	FALSE
"123" != 123	~>	FALSE	"123" !== 123	~>	TRUE
"1.23e2" == 123	~>	TRUE	1.23e2 === 123	~>	FALSE
"1.23e2" == "12.3e1"	~>	TRUE	"1.23e2" === "12.3e1"	~>	FALSE
5 == TRUE	~>	TRUE	5 === TRUE	~>	FALSE

Comparison operators

Type juggling also plays a role in the way PHP comparison operators work:

$expr1 < expr2$	Less than	TRUE iff $expr1$ is strictly less than $expr2$ after type juggling
$expr1 > expr2$	Greater than	TRUE iff $expr1$ is strictly greater than $expr2$ after type juggling
$expr1 <= expr2$	Less than	TRUE iff $expr1$ is less than or equal to $expr2$
$expr1$		$expr2$

<https://eduassistpro.github.io>

'35.5' > 35	~	TRUE	'35	TRUE
'ABD' > 'ABC'	~	TRUE	'AB	TRUE
'1.23e2' > '12.3e1'	~	FALSE	'1.	TRUE
"F1" < "GO"	~	TRUE	"F1	TRUE
TRUE > FALSE	~	TRUE	TRUE >= FALSE	~
5 > TRUE	~	FALSE	5 >= TRUE	~

Revision

Read

- Chapter 3: Introduction to PHP

Assignment Project Exam Help

of

R. Nixo

Learni

<https://eduassistpro.github.io>

O'Reilly, 2009.

Also read [Add WeChat edu_assist_pro](#)

- <http://uk.php.net/manual/en/language.types.intro.php>
- <http://uk.php.net/manual/en/language.types.type-juggling.php>
- <http://uk.php.net/manual/en/language.operators.comparison.php>
- <http://uk.php.net/manual/en/types.comparisons.php>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 10: PHP (Part 2)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

20 Scalar types

Integers and Floating-point numbers

Exceptions and error handling

Bo

Stri

<https://eduassistpro.github.io>

21 Compound types

Arrays

Foreach-loop

Array functions

Add WeChat edu_assist_pro

22 Printing

Integers and Floating-point numbers

- PHP distinguishes between

- integer numbers

0

2012

-40

1263978

- floating-point numbers

1.23

256.0

-1.2e-9

2.34e-10

Assignment Project Exam Help

- PHP supports a wide range of pre-defined mathematical functions

- abs(

- ceil

- floo

- roun

- log(*number* [, *base*])

- log

- rand(*min*, *max*)

- gen

- sqrt(*number*)

- squ

<https://eduassistpro.github.io>

Add WeChat `edu_assist_pro`

- PHP provides a range of pre-defined number constants including

- M_PI

- 3.14159265358979323846

- NAN

- 'not a number'

- INF

- 'infinity'

Integers and Floating-point numbers: NAN and INF

The constants NAN and INF are used as **return values** for some applications of mathematical functions that do not return a number

- `log(0)` returns -INF ('negative infinity')
- `sqrt(-1)` returns NAN ('not a number')

In contrast,

- `1/0`
- `0/0` returns **FALSE** and produces an error message

and execution of the script continues!

Add WeChat edu_assist_pro

In PHP 7

- `1/0` returns **INF** and produces an error message
- `0/0` returns **NAN** and produces an error message

and execution of the script continues!

Integers and Floating-point numbers: NAN and INF

NAN and INF can be compared with each other and other numbers using equality and comparison operators:

`NAN == NAN` FALSE

`INF == INF` FALSE

`NAN < NAN`

`NAN < INF`

`NAN < 1`

`NAN == NAN` FALSE

`INF === INF` TRUE

`NAN < INF`

`INF === 1`

`NAN < 1`

`NAN == 1` FALSE

`INF == 1` FALSE

E

SE

<https://eduassistpro.github.io>

In PHP 5.3 and earlier versions, `INF == INF`

In PHP 5.4 and later versions, `INF == INF`

Add WeChat `edu_assist_pro`

Integers and Floating-point numbers: NAN and INF

- PHP provides three functions to test whether a value is or is not NAN, INF or -INF:

• `bool is_nan(value)`
 returns TRUE iff *value* is NAN

- `bool is_infinite(value)`
 retu
- `bo`
 retu

<https://eduassistpro.github.io>

- In conversion to a boolean value,

both NAN and INF are converted to TRUE

- In conversion to a string,

NAN converts to 'NAN' and INF converts to 'INF'

Add WeChat edu_assist_pro

Exceptions and error handling

PHP distinguishes between `exceptions` and errors

- A possible way to perform `exception handling` in PHP is as follows:

```
try { ... fun code here ... }
} catch (Exception $e) {
    ... handle the exception here using $e // catch
}
```

- Errors <https://eduassistpro.github.io/>
(Divi)

One possible approach is to let the error handling function turn `errors` into `exceptions`.

```
Add WeChat edu_assist_pro
function exception_error_handler($errno, $errstr,
    $errfile, $errline ) {
    throw new ErrorException($errstr, $errno,
        0, $errfile, $errline); }
set_error_handler("exception_error_handler");
```

<http://www.php.net/manual/en/class.errorexception.php>

Booleans

- Unlike Perl, PHP does have a boolean datatype with constants `TRUE` and `FALSE` (case insensitive)

- PHP offers the same short-circuit boolean operators as Java and Perl:
`&&` (conjunction) `||` (disjunction) `!` (negation)

- Alte
- Ho

ectively
<https://eduassistpro.github.io>

- The truth tables for these operators are the same as for Perl
- Remember that `&&` and `||` are not commutative
 - ($A \&\& B$) is not the same as ($B \&\& A$)
 - ($A || B$) is not the same as ($B || A$)

Add WeChat edu_assist_pro

Type conversion to boolean

When **converting to boolean**, the following values are considered **FALSE**:

- the boolean **FALSE** itself
- the integer 0 (zero)
- the float 0.0 (zero)
- the em
- an arra
- an obje
- the special type **NULL** (including **unset**)
- SimpleXML objects created from empty tags

Every other value is considered **TRUE** (including any resource)

Strings

- PHP supports both **single-quoted** and **double-quoted** strings
- PHP also supports **heredocs** as a means to specify multi-line strings

The only difference to Perl is the use of `<<<` instead of `<<` in their definition:

`<<< id`

`here docu`

`identi`

<https://eduassistpro.github.io>

`• ide`

`• identifier` might also be surrounded by single-
making the string a `nowdoc` in PHP terminology.

Add WeChat edu_assist_pr

```
print '<html>
<head><title>Multi-line『String </title></head>';
print <<<EOF
<body>Some text</body>
</html>
EOF;
```

Strings

- Variable interpolation is applied to double-quoted strings (with slight differences to Perl)

The string concatenation operator is denoted by ' .' (as in Perl)

- Instead of Perl's string multiplication operator 'x' there is stri
- There is

<https://eduassistpro.github.io/>

```
$title  = "String"
$string = "<p>I shall not repeat myself.<p>\n";
print "<!DOCTYPE html>\n<html><head><title>$title</ti
</head><body >' . str_repeat($string, 6) . '</body ></htm
<!DOCTYPE html>
<html><head><title>String Multiplication</title>
</head><body ><p>I shall not repeat myself.<p>
<p>I shall not repeat myself.<p>
<p>I shall not repeat myself.<p>
</body ></html >
```

Add WeChat edu_assist_pro

Arrays

- PHP only supports associative arrays (hashes), simply called arrays
- PHP arrays are created using the array construct or, since PHP 5.4:

```
array(key => value, ...)  
[key => value, ...]
```

where

includ

<https://eduassistpro.github.io>

```
$arr1 = [1 => "Pe  
$arr2 = array(200846369 => array("name" => "Jan\u00f8 Olsen",
```

Add WeChat edu_assist_pro

- The size of an array can be determined using the :

```
int count(array [, mode])
```

```
print count($arr1); // prints 3  
print count($arr2); // prints 1  
print count($arr2,1); // prints 4
```

Arrays

- It is possible to omit the keys when using the `array` construct:

```
$arr3 = array("Peter", "Paul", "Mary");
```

The values given in `array` will then be associated with the natural numbers 0, 1, ...

- All the `keys` of `array`

~ returns a natural number-indexed array containing the keys of `$array1`

- All the `values` of an array can be retrieved using `array_values($array1)`

~ returns a natural number-indexed array containing the values stored in `$array1`

Arrays

- An individual array element can be accessed via its `key`
- Accessing an `undefined key` produces an error message and returns `NULL`

Assignment Project Exam Help

```
$arr1 = array(1 => "Peter", 3 => 2009, "a"=> 101);

print '';
'a': 101
print '',
'b': // $arr1["b"] returns NULL
$arr1['b'] = 102;
print "'b':".$arr1["b"]."\n";
'b': 102
```

Add WeChat edu_assist_pro

Arrays

- PHP allows the construct

```
$array[] = value;
```

PHP will determine the maximum value M among the integer indices in $\$array$ and use the key $K = M + 1$; if there are no integer indices in $\$arr$

$\sim au$

```
$arr4[] = ;  
$arr4[] = 42; // 1 => 42  
$arr4[] = 33; // 2 => 33
```

Add WeChat edu_assist_pro

- A key-value pair can be removed from an array using the `unset` function:

```
$arr1 = array(1 => "Peter", 3 => 2009, "a" => 101);  
unset($arr1[3]); // Removes the pair 3 => 2009  
unset($arr1); // Removes the whole array
```

Arrays: foreach-loop

- PHP provides a **foreach-loop** construct to 'loop' through the elements of an array
- Syntax and semantics is slightly different from that of the corresponding construct in Perl

```
foreach ( s
    foreach ( statement )
```

<https://eduassistpro.github.io>

- *array* is an array expression
- *\$key* and *\$value* are two variables storing a *key-value pair* from the *array* at each iteration of the **foreach-loop**
- We call *\$value* the **foreach-variable**
- **foreach** iterates through an array in the order in which elements were defined

Arrays: foreach-loop

`foreach` iterates through an array in the order in which elements were defined

Example 1

```
foreach($array("Peter", "Paul", "Mary") as $key => $value)
    print "The array maps $key to $value\n";
```

The array maps 0 to Pe

The array maps 1 to Pa

The array maps 2 to Ma

<https://eduassistpro.github.io>

Example 2:

```
$arr5[2] = "Marry";
$arr5[0] = "Peter";
$arr5[1] = "Paul";
// 0 => 'Peter', 1 => 'Paul', 2 => 'Marry'
foreach ($arr5 as $key => $value)
    print "The array maps $key to $value\n";
```

The array maps 2 to Mary

The array maps 0 to Peter

The array maps 1 to Paul

Arrays: foreach-loop

Does changing the value of the `foreach-variable` change the element of the list that it currently stores?

Example 3:

```
$arr6 = array("name" => "Peter", "year" => 2009);
```

```
foreach ($ar  
    print  
        $valu  
}
```

```
print "\n";
```

```
foreach($arr6 as $key => $value)  
    print "The array now maps $key to $value\n";
```

The array maps name to Peter

The array maps year to 2009

The array now maps name to Peter

The array now maps year to 2009

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Arrays: foreach-loop

- In order to modify array elements within a **foreach-loop** we need use a reference

```
foreach (array as &$value) {  
    // statement  
    unset($value);
```

```
foreach (array as $value) {  
    // statement  
    unset($value);
```

- In the code schemata above, `&$value` is a variable located in the same location as an array element
- Note that PHP does not allow the `key` to be a reference
- The `unset` statement is important to return `$value` to being a 'normal' variable

Arrays: foreach-loop

In order to modify array elements within a **foreach-loop** we need use a **reference**

Example:

```
$arr6 = array("name" => "Peter", "year" => 2009);
foreach ($ar
    print "
    $value
}
unset($value); // Remove the reference from $value
print "\n";
foreach ($arr6 as $key => $value)
    print "The array maps $key to $value\n";
```

The array maps name to Peter

The array maps year to 2009

The array now maps name to Peter - modified

The array now maps year to 2009 - modified

Array functions

PHP has no `stack` or `queue` data structures,
but has `stack` and `queue` functions for `arrays`:

- `array_push($array, value1, value2, ...)`

appends one or more elements at the end of the end of an array variable;
return

- `array_pop($array)`
extracts the last element of an array and returns it
- `array_shift($array)`
shift extracts the first element of an array and returns it
- `array_unshift($array, value1, value2, ...)`
inserts one or more elements at the start of an array variable
returns the number of elements in the resulting array

Note: `$array` needs to be a `variable`

Printing

In PHP, the default command for generating output is `echo`

Assignment Project Exam Help

- Out
- No p
- Mor

<https://eduassistpro.github.io>

Additionally, PHP also provides the functions

- `int print()`
- Only one argument is allowed!
- Outputs its argument
 - Returns value 1
 - Parentheses can be omitted

Printing

- `string sprintf(format, arg1, arg2, ...)`

- Returns a string produced according to the formatting string *format*

Parentheses are necessary

Assignment Project Exam Help

See <http://www.php.net/manual/en/function.strftime.php>
for det

- `int` <https://eduassistpro.github.io/>

- Produces output according to *format*

- Parentheses are necessary

- Returns the length of the outputted string

Add WeChat `edu_assist_pro`

- **Important:** In contrast to Perl, a PHP array cannot take the place of a list of arguments

```
printf("%2d apples %2d oranges\n", array(5,7));
```

produces an error message

Printing

- `string vsprintf(format, array)`

- Returns a string produced according to the formatting string `format`
Identical to `printf` but accepts an `array` as argument
- Parentheses are necessary

- `int`

<https://eduassistpro.github.io/>

- Pro

- Identical to `printf` but accepts an `array` as argument

- Parentheses are necessary

Add WeChat edu_assist_pro

```
vprintf ("%2d apples %2d oranges\n", array(5,7));
```

```
5 apples 7 oranges
```

Revision

Read

- Chapter 6: PHP Arrays

of

Assignment Project Exam Help

R. Nixo

Learni <https://eduassistpro.github.io>

O'Reilly, 2009.

- <http://uk.php.net/manual/en/language.t>
- <http://uk.php.net/manual/en/language.t>
- <http://uk.php.net/manual/en/language.types.float.php>
- <http://uk.php.net/manual/en/language.types.string.php>
- <http://uk.php.net/manual/en/language.types.array.php>
- <http://uk.php.net/manual/en/control-structures.foreach.php>

Add WeChat edu_assist_pro

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 11: PHP (Part 3)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

23 Special types

NULL

Resources

24 Control structures

Conditional statements

Switch

While

For

<https://eduassistpro.github.io>

25 Functions

Defining a function

Calling a function

Variables

Functions and HTML

Variable-length argument lists

Add WeChat edu_assist_pro

26 PHP libraries

Include/Require

NULL

- `NULL` is both a special type and a value

- `NULL` is the only value of type `NULL`.

and the name of this constant is case-insensitive

- A `variable` has both type `NULL` and value `NULL` in the following three situations

1 `https://eduassistpro.github.io`
2
3 The variable has been `unset` using the `unset` operation

- There are a variety of functions that can be used to test whether a variable is `NULL`, including:

- `bool isset($variable)`

TRUE iff `$variable` exists and does not have value `NULL`

- `bool is_null(expr)`

TRUE iff `expr` is identical to `NULL`

NULL

Warning: Using `NULL` with `==` may lead to counter-intuitive results

```
$d = array();
echo var_dump($d), "\n";
array(0) {
}
echo 'is
is_nul
echo '$d
$d === null: FALSE
echo '$d==null:', ($d == null) ? "TRUE\n": "FALS
$d == null: TRUE
```

Type juggling means that an empty array is (loosely) equal to `NULL`

L

Resources

A **resource** is a reference to an external resource and corresponds to a Perl **filehandle**

Assignment Project Exam Help

`• resource fopen('filename', mode)`
Returns a file pointer resource for *filename* access using *mode* on success

Mod				
'r'				
'r+'	read/write file			
'w'	write file	yes	y	
'w+'	read/write file	yes	y	
'a'	append file	yes		
'a+'	read/append file	yes		
'x'	write file	yes		
'x+'	read/write file	yes		

See <http://www.php.net/manual/en/resource.php> for further details

Resources

- `bool fclose(resource)`

- Closes the resource

Returns TRUE on success

- `string fgets(resource [, length])`

- Ret

retu

- Wit

have been read, or a newline or on EOF (whichever com

- `string fread(resource, length)`

- Returns `length` characters read from `re`

```
$handle = fopen('somefile.txt', 'r');
while ($line = fgets($handle)) {
    // processing the line of the file
}
fclose($handle);
```

Resources

- `int fwrite(resource, string [, length])`

- Writes a string to a resource

If `length` is given, writing stops after `length` bytes have been written or the end of `string` is reached, whichever comes first

- `int`

- Wri
- Iden

<https://eduassistpro.github.io>

- `int vfprintf (resource, format,`

- Writes the elements of an array to a resource in the given f

- Identical to `vprintf` with output to `res`

```
$handle = fopen('somefile.txt', 'w');
fwrite($handle,"Hello World!".PHP_EOL); // 'logical newline'
fclose($handle);
```

In contrast to Perl, in PHP \n always represents the character with ASCII code 10 not the platform dependent newline ↪ use `PHP_EOL` instead

Control structures: conditional statements

The general format of **conditional statements** is very similar but not identical to that in Java and Perl:

```
if (condition) {  
    statements  
} elseif (condition) {  
    sta  
} else {  
    sta  
}
```

<https://eduassistpro.github.io>

- the **elseif-clauses** is optional and there can be more than one.
Note: elseif instead of else-if!
- the **else-clause** is optional but there can be at most one
- in contrast to Perl, the **curly brackets** can be omitted if there is only a **single statement** in a clause

Control structures: conditional statements/expressions

- PHP allows to replace **curly brackets** with a **colon** : combined with an **endif** at the end of the statement:

*if (condition) {
 statements
}*

*elseif (condition) :
 s
else:
 s
endif*

https://eduassistpro.github.io

This also works for the **switch** statement in PHP

However, this syntax becomes difficult to parse when nested conditional statements are used and it

- PHP also supports **conditional expressions**

condition ? if_true_expr : if_false_expr

Control structures: switch statement

A **switch statement** in PHP takes the following form

```
switch (expr) {  
    case expr1:  
        statements  
    case expr2:  
        statements  
    ...  
    default:  
        statements  
    }  
}
```

- there can be arbitrarily many **case**-clauses
- the **default**-clause is optional but there can be at most one

<https://eduassistpro.github.io/>

corresponding clause

if none of *expr1*...
then the **default**-clause

- **break** ‘breaks out’ of the switch statement
- if a clause does not contain a **break** command, then execution moves to the next clause

Control structures: switch statement

Example:

```
switch ($command) {  
    case "North":  
        $y += 1; break;  
    case "South":  
        $y - 1  
    case "West":  
        $x -= 1  
    case "East":  
        $x += 1; break;  
    case "Search":  
        if (($x = 5) && ($y = 3))  
            echo "Found a treasure!\n";  
        else  
            echo "Nothing here\n";  
        break;  
    default:  
        echo "Not a valid command\n"; break;  
}
```

Assignment Project Exam Help
<https://eduassistpro.github.io>

Control structures: switch statement

Not every `case`-clause needs to have associated statements

Example:

```
switch ($month) {  
    case 1:    case 3:    case 5:    case 7:  
    case 8:    case 10:   case 12:  
        $d  
        br  
    case 4:    case 6:    case 9:    case 11:  
        $days = 30;  
        break;  
    case 2:  
        $days = 28;  
        break;  
    default:  
        $days = 0;  
        break;  
}
```

Add WeChat edu_assist_p

Control structures: while- and do while-loops

- PHP offers **while-loops** and **do while-loops**

```
while (condition) {  
    statement  
}
```

Assignment Project Exam Help

- As usual, **curly brackets** can be omitted if the loop consists of only one statement

Add WeChat edu_assist_pro

Example:

```
// Compute the factorial of $number  
$factorial = 1;  
do {  
    $factorial *= $number--;  
} while ($number > 0);
```

Control structures: for-loops

- for-loops in PHP take the form

```
for (initialisation; test; increment) {  
    ...  
}
```

Assignment Project Exam Help

- Again consists of three parts:
 - In PHP statement, separated by commas instead of semicolons

Example:

Add WeChat edu_assist_pro

```
for ($i = 3, $j = 3; $j >= 0; $i++, $j--)  
    echo "$i - $j - ", $i*$j, "\n";
```

```
3 - 3 - 9  
4 - 2 - 8  
5 - 1 - 5  
6 - 0 - 0
```

Control structures: break and continue

- The `break` command can also be used in while-, do while-, and for-loops and discontinues the execution of the loop

```
while ($value = array_shift($data)) {  
    $written = fwrite($resource, $value);  
    if (!$written) break;  
}
```

- The `continue` command loops back to the top of the loop and continues the loop again.

```
for ($x = -2; $x <= 2; $x++) {  
    if ($x == 0) continue;  
    printf("10 / %d = %d\n", $x, 10/$x);  
}
```

```
10 / -2 = -5  
10 / -1 = -10  
10 / 1 = 10  
10 / 2 = 5
```

Functions

Functions are defined as follows in PHP:

```
function identifier($param1, &$param2, ...) {  
    statements  
}
```

Assignment Project Exam Help

- Function identifiers should be meaningful
- Functions should have parameters
- The function name must be followed by parentheses
- A function has zero or more parameters that are passed by value
- Parameters can be given a default value using:
 $\$param = const_expr$
- When using default values, any defaults must be on the right side of any parameters without defaults

Functions

Functions are defined as follows in PHP:

```
function identifier($param1, &$param2, ...) {  
    statements  
}
```

Assignment Project Exam Help

- The `ret`

The `return` statement can be used to return a `value` the return value of the function

- The `return value` does not have to be scalar value
- A function can contain more than one `return` statement
- Different return statements can return values of different types

Add WeChat `edu_assist_pro`

Calling a function

A function is **called** by using the function name followed by a list of **arguments** in parentheses

```
function identifier($param1, &$param2, ...){  
}  
... ide
```

- The list of arguments must have the same number of parameters as the list of parameters
- If it is shorter, then **default values** must have been specified for the remaining parameters without corresponding argument

Example:

```
function sum($num1,$num2) {  
    return $num1+$num2;  
}  
echo "sum: ",sum(5,4),"\n";  
$sum = sum(3,2);
```

Add WeChat edu_assist_pro

Variables

PHP distinguishes three categories of variables:

- Local variables are only accessible in the part of the code in which they are introduced

Assignment Project Exam Help

- Global

- Static variables retain their value between separate calls of the function

By default, variables in PHP are local but not static
(Variables in Perl are by default global)

Add WeChat edu_assist_pro

PHP functions: Example

```
function bubble_sort($array) {  
    // $array, $size, $i, $j are all local  
    if (!is_array($array))  
        trigger_error('Argument must be an array\n', E_USER_ERROR);  
    $size = count($array);  
    for ($i=0  
        for ($  
            return $array;  
    }  
  
function swap(&$array, $i, $j) {  
    // swap expects a reference (to an array)  
    $tmp = $array[$i];  
    $array[$i] = $array[$j];  
    $array[$j] = $tmp;  
}
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

PHP functions: Example

```
function bubble_sort($array) {  
    ... swap($array, $j, $j+1); ...  
    return $array;  
}
```

Assignment Project Exam Help

```
function swa  
{  
    $tmp = $arr  
    $array  
    $array
```

```
$array = array(2,4,3,9,6,8,5,1);  
echo "Before sorting ", join(" ", $array), "\n";  
$sorted=bubble_sort($array);  
echo "After sorting ", join(" ", $array), "\n";  
echo "Sorted array ", join(" ", $sorted), "\n";
```

Before sorting 2, 4, 3, 9, 6, 8, 5, 1

After sorting 2, 4, 3, 9, 6, 8, 5, 1

Sorted array 1, 2, 3, 4, 5, 6, 8, 9

Functions and global variables

- A variable is declared to be **global** using the keyword **global**

```
function echo_x($x) {  
    echo $x; "";  
    global $x;  
    echo $x;  
}
```

<https://eduassistpro.github.io/>

- an otherwise **local** variable is made accessible outside its scope using **global**
- all **global** variables with the same name refer to the same storage location/data structure
- an **unset** operation removes a specific variable, but leaves other (global) variables with the same name unchanged

PHP functions and Global variables

```
function modify_or_destroy_var($arg) {  
    global $x, $y;  
    if (is_bool($arg) && !$arg) { $x = $x + $y; }  
    if (is_bool($arg) && !$arg) { unset($x); echo $x; }  
}  
  
$x = 2; $y = 3; $z = 4;  
echo "1: \$x = $x, \$y = $y  
1: $x = 2, $y = 3, $z = 4  
unset($z)  
echo "2: \$x = $x, \$y = $y, \$z = $z\n";
```

PHP Notice: Undefined variable: z in script on line 9

```
2: $x = , $y = 3, $z =  
modify_or_destroy_var(false);  
echo "3: \$x = $x, \$y = $y\n";  
3: $x = 6, $y = 3
```

```
modify_or_destroy_var(true);  
echo "4: \$x = $x, \$y = $y\n";
```

PHP Notice: Undefined variable: x in script on line 4

```
4: $x = 6, $y = 3
```

PHP functions and Static variables

- A variable is declared to be **static** using the keyword **static** and should be combined with the assignment of an initial value (initialisation)

function counter() { static \$count = 0; return \$count++; }

~ static variables are initialised only once

```
1 function counter() { static $count = 0; return $count++; }
2 $count = 5
3 echo "1: $count";
4 echo "2: static \$count = ",counter(),"\n";
5 echo "3: static \$count = ",counter(),"\n";
6 echo "4: global \$count = $count\n";
1: global $count = 5
2: static $count = 0
3: static $count = 1
4: global $count = 5
```

Add WeChat edu_assist_pro

Functions and HTML

- It is possible to include **HTML markup** in the body of a function definition
- The **HTML markup** can in turn contain **PHP scripts**
- A call of the function will execute the PHP scripts, insert the output into th

```
<?php  
function print_f  
?>  
<form action="process_form.php" method=POST>  
<label>First Name: <input type="text" name="f" value="<?php echo $fn?>"></label><br>  
<label>Last Name<b>*</b>:<input type="text" name="l" value="<?php echo $ln?>"></label><br>  
<input type="submit" name="submit" value="Submit"> <input type=reset>  
</form>  
<?php  
}  
print_form("Ullrich","Hustadt");  
?>
```

Add WeChat edu_assist_pr

```
<form action="process_form.php" method=POST>  
<label>First Name: <input type="text" name="f" value="Ullrich"></label><br>  
<label>Last Name<b>*</b>:<input type="text" name="l" value="Hustadt"></label><br>  
<input type="submit" name="submit" value="Submit"> <input type=reset>  
</form>
```

Functions with variable number of arguments

The number of arguments in a function call is allowed to exceed the number of its parameters

the parameter list only specifies the minimum number of arguments

- int func_num_args()

return

- mixed func_get_args()

return

- array func_get_args()

returns an array with copies of the arguments passed

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

```
function sum() { // no minimum number of arguments
    if (func_num_args() < 1) return null;
    $sum = 0;
    foreach (func_get_args() as $value) { $sum += $value; }
    return $sum;
}
```

Including and requiring files

- It is often convenient to build up **libraries** of function definitions, stored in one or more files, that are then reused in PHP scripts
- PHP provides the commands `include`, `include_once`, `require` and `require_once` to incorporate the content of a file into a PHP script

```
include '
```

- PHP code:
`<?ph`
- The incorporated content inherits the scope of the line where the `include` command occurs
- If no absolute or relative path is specified, PHP will search:
 - first, in the directories in the `include path` `include_path`
 - second, in the script's directory
 - third, in the current working directory

Add WeChat edu_assist_pro

Including and requiring files

- Several `include` or `require` commands for the same library file results in the file being incorporated several times

→ defining a function more than once results in an error

Assignment Project Exam Help

- Several `include_once` or `require_once` commands for the same library

- If a library file is found, it can only be included once by the requesting script

- If a library file requested by `require` is not found, PHP generates an error and stops execution of the script

Add WeChat edu_assist_pr

PHP Libraries: Example

mylibrary.php

```
<?php
function bubble_sort($array) {
    ...
    swap($array, $j, $j+1);
    ...
    return $array;
}

function swap {
    ...
}
?>
```

example.php

```
<?php
require_once 'mylibrary.php';
$array = array(2,4,3,9,6,8,5,1);
$sorted = bubble_sort($array);
?>
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Revision

Read

- Chapter 4: Expressions and Control Flow in PHP
- Chapter 5: PHP Functions and Objects
- Chapter 7: Practical PHP

of

R. Nixon:

Learnin

O'Reilly, 2009.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

- <http://uk.php.net/manual/en/language.c>
- <http://uk.php.net/manual/en/language.f>
- <http://uk.php.net/manual/en/function.include.php>
- <http://uk.php.net/manual/en/function.include-once.php>
- <http://uk.php.net/manual/en/function.require.php>
- <http://uk.php.net/manual/en/function.require-once.php>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 12: PHP (Part 4)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

27 Web applications

 Overview

 HTML forms

28 Available information and input

 Overview

 PHP

 Server

 Form

29 PHP sessions

 Start a PHP session

 Maintain session data

 End a PHP session

 Session management

 Example

30 Authentication

 Overview

 Example

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Web applications using PHP

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

IBM: Build Ajax-based Web sites with PHP, 2 Sep 2008.

<https://www.ibm.com/developerworks/library/wa-aj-php/> [accessed 6 Mar 2013]

HTML forms

When considering Perl CGI programming we have used HTML forms that generated a **client request** that was handled by a **Perl CGI program**:

```
<form action=
  "http://cgi.csc.liv.ac.uk/cgi-bin/cgiwrap/ullrich/demo"
  method="post">
...
</form>
```

Now we will

```
<form action="http://cgi.csc.liv.ac.uk/~ullrich/demo.php"
  method="post">
...
</form>
```

Add WeChat edu_assist_p

- The PHP script file must be stored in a directory accessible by the web server, for example \$HOME/public_html, and be readable by the web server
- The PHP script file name must have the extension .php, e.g. demo.php

Information available to PHP scripts

- Information about the **PHP environment**
- Information about the **web server** and **client request**
- Information stored in **files** and **databases**
- Form data
- Cookies
- Miscellaneous

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

- **string date(*format*)**
returns the current date/time presented according to
for example `date('H:i:s j, jS F Y')`
results in `12:20 Thursday, 8 March 2012`

(See <http://www.php.net/manual/en/function.date.php>)

- **int time()**
returns the current time measured in the number of seconds
since January 1 1970 00:00:00 GMT

PHP environment

- `phpinfo()` displays information about the PHP installation and EGPCS data (Environment, GET, POST, Cookie, and Server data) for the current client request
- `phpinfo(part)` displays selected information

```
<html><he  
<?php  
    phpinf  
    phpinf  
?>  
</body></html>
```

<https://eduassistpro.github.io/>

<http://cge.cscliv.ac.in/~mlrich/COMP28>

Add WeChat edu_assist_pro

<code>INFO_GENERAL</code>	The configuration, path to web server
<code>INFO_CONFIGURATION</code>	Local and master values for PHP directives
<code>INFO_MODULES</code>	Loaded modules
<code>INFO_VARIABLES</code>	All EGPCS data

Manipulating the PHP configuration

The following functions can be used to access and change the configuration of PHP from within a PHP script:

Assignment Project Exam Help

- `array ini_get_all()`
 - returns all the registered configuration options
- `string stri`
 - retu <https://eduassistpro.github.io>
- `string ini_set(option, value)`
 - sets the value of the given configuration option to a new value
 - the configuration option will keep this new value during execution and will be restored afterwards
- `void ini_restore(option)`
 - restores a given configuration option to its original value

Server variables

The `$_SERVER` array stores information about the web server and the client request

Similar to `%ENV` for Perl CGI programs

```
<html><head></head><body>
<?php
echo 'Server: ' . $_SERVER['SERVER_NAME'];
echo 'Remote: ' . $_SERVER['REMOTE_ADDR'];
echo 'Client: ' . $_SERVER['HTTP_USER_AGENT'];
echo 'Request method: ' . $_SERVER['REQUEST_METHOD'];
?></body></html>
```

<http://cgi.csc.liv.ac.uk/~mlrich/COMP288>

Server software: Apache/2.2.22 (Fedora)

Remote address: 10.128.0.215

Client browser: Mozilla/5.0 ... Chrome/41.0.2272.53 ...

Request method:

See <http://php.net/manual/en/reserved.variables.server.php> for a list of keys

Form data

- Form data is passed to a PHP script via the three arrays:

`$_POST`

Data from POST client requests

`$_GET`

Data from GET client requests

`$_REQUEST`

Combined data from POST and GET client requests

Assignment Project Exam Help

~ Ac
usi

<https://eduassistpro.github.io>

```
<form action="process.php" method="post">
<label>Enter your user name:<br>
    <input type="text" name="username"></label><br>
<label>Enter your full name:<br>
    <input type="text" name="fullname"></label><br>
<input type="submit" value="Click for response"></form>
```

`$_REQUEST['username']` Value entered into field with name 'username'

`$_REQUEST['fullname']` Value entered into field with name 'fullname'

Forms in PHP: Example (1)

- Create a web-based system that asks the user to enter the URL of a file containing bibliographic information
- Bibliographic information will have the following form:

```
@entry{  
    name=  
    name=  
    title=  
}  
@entry{  
    name={Andreas Schoknecht},  
    name={Eva Eggerling},  
    title={No End in Sight?}  
}
```

Add WeChat edu_assist_pr

- The system should extract the names, count them, and create a table of names and their frequency, ordered from most frequent to least frequent

Forms in PHP: Example (1)

```
extract_names.php
<!DOCTYPE html>
<html><head><title>Name Extraction</title></head><body>
<?php
require 'extract_name.php';
if (isset($_SERVER['REQUEST_METHOD'])) &&
    $_SERVER['REQUEST_METHOD'] == 'POST' &&
    isset(
        $extract
    )
    echo "<p>" . extract_name($_POST['url']) . "</p>";
} else {
    echo <<<FORM
<form method="post">
    <label>Enter a URL:<br>
    <input type="text" name="url" size="100" value="http://cgit.csc.liv.ac.uk/~ullrich/<br>
    </label><br><br>
    <input type="submit" value="Extract Names">
</form>
FORM;
}
?>
</body></html>
http://cgit.csc.liv.ac.uk/~ullrich/COMP284/examples/extract_names.php
```

Assignment Project Exam Help
<https://eduassistpro.github.io>

Forms in PHP: Example (1)

extraction.php

```
<?php
function extract_names($url) {
$text = file_get_contents($url);
if ($text === false)
    return "ERROR: INVALID URL!";
else {
$corr
if ($cor
$count = array_count_values($matches[1]);
arsort($count);
foreach ($count as $name => $number) {

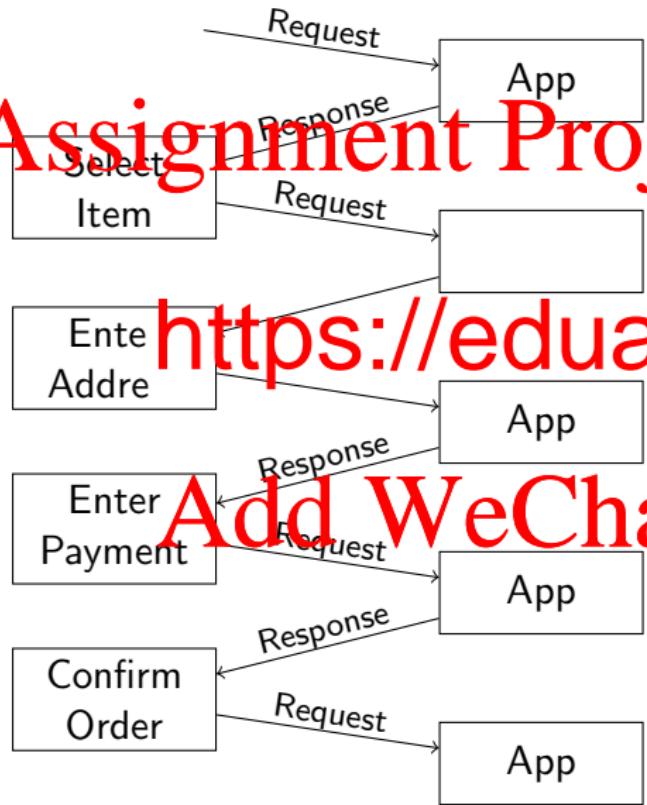

|        |          |
|--------|----------|
| \$name | \$number |
|--------|----------|


```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/extraction.php>

Web Applications Revisited

Assignment Project Exam Help



- An **interaction** between a user and a server-side web application often requires a **sequence** of **requests** and **responses**

ation

uests

.



data needs to be transferred from one execution of the application to the next

Transfer of Data: Example

- Assume for a sequence of requests we do **not** care whether they come from the same user or different users
- Then **hidden inputs** can be used for the transfer of data from one request / page to the next

Assignment Project Exam Help

form1.php

```
<form action
      <label>
</form>
```

<https://eduassistpro.github.io>

form2.php

```
<form action="process.php" method="post">
    <label>Address:</label> <input type="text" name="address"></label>
    <input type='hidden' name='name'
           value="<?php echo $_REQUEST['name'] ?>">
</form>
```

process.php

```
<?php
    echo $_REQUEST['name'];
    echo $_REQUEST['address'];
?>
```

Sessions

- By default, HTML and web servers do not keep track whether several client requests come from the same user or different users
- This is a process that spans several pages, for example placing an order, requires additional mechanisms

- Session

specific

<https://eduassistpro.github.io/>

- Sessions are often linked to user authentication but session can be used without user authentication, for example, eCom
a 'shopping basket' without requiring user auth

Add WeChat edu_assist_pro

However, sessions are the mechanism that is typical
deny access to web pages based on a user having been authenticated

Sessions

- Servers keep track of a user's sessions by using a **session identifier**, which
 - is generated by the server when a session starts and
 - is then used by the browser when the user requests a page from the server

Assignment Project Exam Help

The **ses**

sessio

<https://eduassistpro.github.io>

- In addit

relate to a user and her session (**session data**), for exa

the **items** of an order

Add WeChat edu_assist_pr

- Sessions only store information temporarily

If one needs to preserve information between visits by the same user, one needs to consider a method such as using a **cookie** or a database to store such information

Cookies

Browser

Server

```
GET /index.html HTTP/1.1  
Host: intranet.csc.liv.ac.uk
```

Browser

Server

```
HTTP/1.0 200 OK  
Content-type: text/html  
Set-Cookie: name1=value1
```

Browser

Server

```
Host: intranet.csc.liv.ac.uk  
Cookie: name1=value1; name2=  
Accept: */*
```

Browser

Server

```
HTTP/1.0 200 OK  
Content-type: text/html  
Set-Cookie: name1=value3  
Set-Cookie: name2=value4; Expires= Fri, 21 Mar 2014, 14:00 GMT  
Set-Cookie: name3=value5; Expires= Fri, 28 Mar 2014, 20:00 GMT  
(content of teaching.html)
```

Wikipedia Contributors: HTTP Cookie. Wikipedia, The Free Encyclopedia, 5 March 2014 20:50.
http://en.wikipedia.org/wiki/HTTP_cookie [accessed 6 Mar 2014]

PHP sessions

Sessions proceed as follows

- ① Start a PHP session

- `bool session_start()`
- `string session_id([id])`
- `bo`

- ② Mai

- `bo`
- `$_SESSION` array
- `bool isset($_SESSION[key])`
- (interacting with a database)

- ③ End a PHP session

- `bool session_destroy()`
- `void session_unset()`
- `bool setcookie(name, value, expires, path)`

Start a session

- `bool session_start()`

- creates a session

- creates a session identifier `session_id()`, when a session is created

- sets up `$_SESSION` array that stores session variables and session data

- the fu

- or ou

<https://eduassistpro.github.io>

- `stri`

- get or set the `session id` for the current session

- the constant `SID` can also be used to retrieve the current `session id` as a string suitable for adding to URLs

- `string session_name([name])`

- returns the name of the current session

- if a name is given, the current session name will be replaced with the given one and the old name returned

Start a PHP session

- bool session_regenerate_id([*delete_old*])
 - replaces the current session id with a new one
 - by default keeps the current session information stored in \$SESSION
 - if the optional boolean argument is *TRUE*, then the current session info
- ~ re
bei <https://eduassistpro.github.io/>

```
<?php
    session_start();
    echo "Session id: ", session_id(), "<br />";
    echo "Session name: ", session_name(), "<br />";

    session_regenerate_id();
    echo "Session id: ", session_id(),"<br />";      // changed
    echo "Session name: ", session_name(),"<br />"; // unchanged
?>
```

Maintain session data

- `bool session_start()`
 - resumes the current session based on a session identifier passed via a GET or POST request, or passed via a cookie
 - restores `session variables` and `session data` into `$_SESSION`
 - the fu
or ou
- `$_SE`
 - an associative array containing `session variables` a
 - you are responsible for choosing `keys` (`session varia`) and maintaining the associated `values` (`session dat`)
- `bool isset($_SESSION[key])`
returns TRUE iff `$_SESSION[key]` has already been assigned a value

Maintain session data

- `bool session_start()`
- `$_SESSION` array
 - `bool isset($_SESSION[key])`

Assignment Project Exam Help

```
<?php
// Counting the number of requests
// Each web page counts as one request
session_start();
if (!isset($_SESSION['requests']))
    $_SESSION['requests'] = 1;
else
    $_SESSION['requests']++;
echo "#Requests in this session so far: ",
    $_SESSION['requests'], "<br />\n";
?>
```

<https://eduassistpro.github.io/>

End a PHP session

- bool `session_destroy()`
 - destroys all of the data associated with the current session
 - it does not unset any of the global variables associated with the session, or unset the session cookie
- void
 - free
- bool
 - defines a cookie to be sent along with the rest of the HTTP
 - must be sent before any output from the script
 - the first argument is the `name` of the cookie
 - the second argument is the `value` of the cookie
 - the third argument is `time` the cookie expires (as a Unix timestamp), and
 - the fourth argument is the `path` on the server in which the cookie will be available

Assignment Project Exam Help

Add WeChat `edu_assist_pro`

End a PHP session

- bool session_destroy()
 - destroys all of the data associated with the current session
- void session_unset()
 - frees all session variables currently registered
- bool
 - defi

<https://eduassistpro.github.io>

```
<?php
session_start();
session_unset();
if (session_id() != "" || isset($_COOKIE[session_name()]))
    // force the cookie to expire
    setcookie(session_name(), session_id(), time() -2592000, '/');
session_destroy();
?>
```

Add WeChat edu_assist_pro

Note: Closing your web browser will also end a session

More on session management

The following code tracks whether a session is active and ends the session if there has been no activity for more than 30 minutes

```
if (isset($_SESSION['LAST_ACTIVITY']) &&
    time() - $_SESSION['LAST_ACTIVITY'] > 1800) {
    // last request was more than 30 minutes ago
    session_destroy(); // destroy session data in storage
    session_start();
    if (session_status() == PHP_SESSION_NONE) {
        session_start();
    }
} else {
    // update last activity
    $_SESSION['LAST_ACTIVITY'] = time();
}
```

The following code generates a new session identifier every 30 minutes

```
if (!isset($_SESSION['CREATED'])) {
    $_SESSION['CREATED'] = time();
} else if (time() - $_SESSION['CREATED'] > 1800) {
    // session started more than 30 minutes ago
    session_regenerate_id(true);
    $_SESSION['CREATED'] = time();
}
```

<http://stackoverflow.com/questions/520237/how-do-i-expire-a-php-session-after-30-minutes>

PHP sessions: Example

mylibrary.php:

```
<?php
session_start();
function destroy_session_and_data() {
    session_destroy();
}
function count_requests() {
    if (!isset($_SESSION['requests'])) {
        $_SESSION['requests'] = 1;
    } else {
        $_SESSION['requests']++;
    }
    return $_SESSION['requests'];
}
?>
```

Assignment Project Exam Help
<https://eduassistpro.github.io>

PHP sessions: Example

page1.php:

```
<?php
    require_once 'mylibrary.php';
    echo "<html><head></head><body>\n";
    echo "Hello visitor!<br />This is your page request no ";
    echo count_requests()." from this site.<br />\n";
    echo '<a href
?>
```

<https://eduassistpro.github.io/>

finish.php:

```
<?php
    require_once 'mylibrary.php';
    destroy_session_and_data();
    echo "<html><head></head><body>\n";
    echo "Goodbye visitor!<br />\n";
    echo '<a href="page1.php">Start again</a></body>';
?>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/page1.php>

PHP and Cookies

Cookies can survive a session and transfer information from one session to the next

```
cmylibrary.php:  
<?php  
session_start();  
function des  
  
function cou  
if (!isse  
    setcookie('requests', 1, time() + 31536000, '/');  
    return 1;  
} else {  
// $_COOKIE['requests'] + 1 would not survive, instead us  
    setcookie('requests', $_COOKIE['requests'] + 1,  
              time() + 31536000, '/'); // valid for 1 year  
    return $_COOKIE['requests'] + 1;  
}  
?  
>
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

PHP Sessions and Authentication

- Sessions are the mechanism that is typically used to allow or deny access to web pages based on a user having been authenticated

• Outline solution:

- We want to protect a page `content.php` from unauthorised use

• Bef

the

• The s

and checks usernames and passwords entered by the user against that database

If the check succeeds, a session variable is set.

- The page `content.php` checks whether this session v

If the session variable is set, the user will see the content of the page

If the session variable is not set, the user is redirected to `login.php`

- The system also provides a `logout.php` page to allow the user to log out again

PHP Sessions and Authentication: Example

Second part of login.php:

```
<!DOCTYPE html>
<html>
<head><title>Login</title></head>
<body>
<h1>Login</h1>
<form action="process_login.php">
<label>Username:</label>
<input name="username" type="text" placeholder="Enter Username" />
</label>
<label>Password:</label>
<input name="password" type="password" placeholder="Enter Password" />
</label>
<input name="submit" type="submit" value="login " />
<span><?php echo $error; ?></span>
</form>
</body>
</html>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/login.php>

PHP Sessions and Authentication: Example

First part of login.php:

```
<?php
session_start();
function checkCredentials($user,$passwd) {
    // Check whether $user and $passwd are non-empty
    // and match an en
}
$error='';
if (isset($_POST['submit'])) {
    if (checkCredentials($_REQUEST['user'],$_REQUEST['passwd'],
        $_SESSION['user'])==$_REQUEST['user']){
        header("location:content.php"); // he
    } else {
        $error = "Username or Password is invalid. Try Again";
    }
}
if (isset($_SESSION['user'])){
    header("location:content.php");
}
?>
```

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

PHP Sessions and Authentication: Example

content.php:

```
<?php
session_start();
if (!is_set($_SESSION['user'])) {
    // User is not logged in, redirecting to login page
    header('Location: login.php');
}
?>
<!DOCTYPE html>
<html>
<head><title>Content that requires login</title></head>
<body>
<h1>Protected Content</h1>
<b>Welcome <i><?php echo $_SESSION['user'] ?></i></b><br />
<b><a href="logout.php">Log Out</a></b>
</body>
</html>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/content.php>

PHP Sessions and Authentication: Example

logout.php:

```
<?php
session_start();
$user = $_SESSION['user'];
session_unset();
session_destroy();
?>
<!DOCTYPE html>
<html>
<head>
<title>Logout</title>
</head>
<body>
<h1>Logout</h1>
<b>Goodbye <i><?php echo $user ?></i></b><br />
<b><a href="login.php">Login</a></b>
</form>
</body>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/logout.php>

Revision

Assignment Project Exam Help

Read

- Chapter 10: Accessing MySQL Using PHP
- Chap
- Chap

of <https://eduassistpro.github.io>

R. Nixon:
Learning PHP, MySQL, and JavaScript.
O'Reilly, 2009.

Add WeChat edu_assist_pro

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 13: PHP (Part 5)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

31 Classes

Defining and Instantiating a Class

Visibility

Class Constants

Static Properties and Methods

De

Inh

Int

Introspection Functions

32 The PDO Class

Introduction

Connections

Queries and Processing of Results

Prepared Statements

Transactions

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Defining and Instantiating a Class

- PHP is an object-oriented language with `classes`
- A `class` can be defined as follows:

```
Class identifier {  
    property_definitions  
    function_definitions  
}
```

- The `class` keyword defines a class.
- The body of a class consists of `property definitions` and `function definitions`.
- The function definitions may include the definition of properties.
- An `object` of a class is created using

```
new identifier(arg1, arg2, ...)
```

where `arg1, arg2, ...` is a possibly empty list of arguments passed to the constructor of the class `identifier`.

A Closer Look at Class Definitions

In more detail, the definition of a `class` typically looks as follows

```
class identifier {  
    # Properties  
    vis $attrib1  
    ...  
    vis $a  
  
    # Constructors  
    function _  
        statements  
    }  
    # Methods  
    vis function method1(p1,...) {  
        statements  
    }  
    vis function methodN(p1,...) {  
        statements  
    }  
}
```

Every instance obj of this class will have attributes `attrib1`, ... and methods

ible

<https://eduassistpro.github.io>

Add WeChat edu_assist_pr

, ...)

is executed

- `vis` is a declaration of the `visibility` of each attribute and method

A Closer Look at Class Definitions

- The pseudo-variable `$this` is available when a method is called from within an object context and is a reference to the calling object
- Inside method definitions, `$this` can be used to refer to the properties and methods of the calling object
- The `ob`
callin

```
class Rectan
{
    protected $height;
    protected $width;

    function __construct($height,$width) {
        $this->width = $width;
        $this->height = $height;
    }
}
```

Add WeChat edu_assist_pro

Visibility

- Properties and methods can be declared as

`public` accessible everywhere

`private` accessible only within the same class

`protected` accessible only within the class itself and by inheriting and parent classes

- For `pro` declar
- For `methods`, a `visibility` declaration is optional
 - ~ by default, `methods` are `public`
- Accessing a `private` or `protected` property / method outside its visibility is a `fatal error`

`protected $protected = 3;`
`protected fu`
`private function`
}
`$v = new Vis();`
`echo $v->public; # prints 1`
`echo $v->private; # Fatal Error`
`echo $v->protected; # Fatal Error`
`echo $v->priFc(); # Fatal Error`
`echo $v->proFc(); # Fatal Error`

Add WeChat [edu_assist_pro](https://eduassistpro.github.io)

Constants

- Classes can have their own **constants** and
constants can be declared to be **public**, **private** or **protected**
→ by default, class constants are public
vis const identifier = value;

Assignment Project Exam Help

- Acces

error

- Class c

and not for each class instance

- Class constants are accessed using the **scope resolution operator**:

Add WeChat edu_assist_pr

```
class MyClass {  
    const SIZE = 10;  
}  
echo MyClass::SIZE;    # prints 10  
$o = new MyClass();  
echo $o::SIZE;          # prints 10
```

Static Properties and Methods

- Class properties or methods can be declared `static`
- Static class properties and methods are accessed (via the class) using the scope resolution operator `::`
- Static class properties cannot be accessed via an instantiated class object
- Static c

```
class Employee {
    static $totalNumber = 0;
    public $name;

    function __construct($name) {
        $this->$name = $name;
        Employee::$totalNumber++;
    }
}

$e1 = new Employee("Ada");
$e2 = new Employee("Ben");
echo Employee::$totalNumber # prints 2
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Destructors

- A class can have a **destructor method `__destruct`** that will be called as soon as there are no other references to a particular object

```
class Employee {  
    static $totalNumber = 0;  
    public $name;  
  
    function __construct($name) {  
        $this->name = $name;  
        Employee::$totalNumber++;  
    }  
    function __destruct() {  
        Employee::$totalNumber--;  
    }  
}  
  
$e1 = new Employee("Ada");  
$e2 = new Employee("Ben");  
echo Employee::$totalNumber # prints 2  
$e1 = null;  
echo Employee::$totalNumber # prints 1
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Inheritance

- In a class definition it is possible to specify one **parent** class from which a class inherits constants, properties and methods:

Assignment Project Exam Help

- The constructor of the parent class is **not** automatically called it must be called
- Inheritance
redecl <https://eduassistpro.github.io>
- The declaration **final** can be used to prevent overriding
- Using **parent::** it is possible to access overridden properties of the parent class
- Using **self::** it is possible to access static properties and methods of the current class

Add WeChat edu_assist_pro

Inheritance: Example

```
class Rectangle {  
    protected $height;  
    protected $width;  
    function __construct($height,$width)  
    {  
        $this->width = $width;  
        $this->height = $height;  
    }  
    function area()  
    {  
        return $this-> height * $this-> width;  
    }  
}  
  
class Square extends Rectangle {  
    function __construct($size)  
    {  
        parent::__construct($size,$size);  
    }  
}  
  
$rt1 = new Rectangle(3,4);  
echo "\$rt1 area = ",$rt1->area(),"\n";  
$sq1 = new Square(5);  
echo "\$sq1 area = ",$sq1->area(),"\n";  
$rt1 area = 12  
$sq1 area = 15
```

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

Interfaces

- Interfaces specify which methods a class must implement without providing an implementation
- Interfaces are defined in the same way as a class with the keyword `class` replaced by `interface`

- All met

- A class can implement multiple interfaces

```
interface Shape {  
    public function area();  
}  
class Rectangle implements Shape {  
    ...  
}
```

Add WeChat edu_assist_pro

Introspection Functions

There are functions for inspecting objects and classes:

`bool class_exists(string class)`

returns TRUE iff a class *class* exists

`class_exists('Rectangle') # returns TRUE`

`string`

returns *t*

`get_c`

`bool`

returns TRUE iff *obj* is an instance of class name

`is_a($sq1, 'Rectangle') # no`

`bool method_exists(object obj, string method)`

returns TRUE iff *obj* has a method named *method*

`method_exists($sq1, 'area') # returns TRUE`

Introspection Functions

There are functions for inspecting objects and classes:

`bool property_exists(object obj, string property)`

returns TRUE iff `obj` has a property named `property`

`property_exists($sql, 'size') # returns FALSE`

`get_o`

returns

mappe

`get_o`

`# returns ["name" => "Ben"]`

`get_class_methods(class)`

returns an array of method names defined for

`get_class_methods('Square')`

`# returns ["__construct", "area"]`

The PDO Class

Assignment Project Exam Help

- The PHP Data Objects (PDO) extension defines an interface for access
- Vario datab <https://eduassistpro.github.io>
 - PDO_MYSQL implements the PDO interface for MySQL
 - PDO_SQLSRV implements the PDO interface for Microsoft Azure SQL Database

Add WeChat edu_assist_pro

Connections

- Before we can interact with a DBMS we need to establish a [connection](#) to it
- A connection is established by [creating an instance](#) of the [PDO class](#)
- The [constructor](#) for the [PDO class](#) accepts arguments that specify the database

```
$pdo = new PDO
```

<https://eduassistpro.github.io>

- Upon successfully establishing a connection, a [PDO instance](#) is created
- The connection remains [active](#) for the lifetime of the script
- Assigning [NULL](#) to the variable storing the [PDO object](#) [closes the connection](#)

```
$pdo = NULL
```

Connections: Example

```
# Connection information for the Departmental MySQL Server
$host      = "mysql";
$user      = "ullrich";
$password = "-----";
$db        = "ullrich";
$charset   = "utf8mb4";
$dsn       = "mysql:host=$

# Useful options
$opt = array(
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES  => false
);

try {
    $pdo = new PDO($dsn,$user,$password,$opt);
} catch (PDOException $e) {
    echo 'Connection failed: ', $e->getMessage();
}
```

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

Queries

- The `query()` method of PDO objects can be used to execute an SQL query

```
$result = $pdo->query($statement)  
$result = $pdo->query("SELECT * FROM meetings")
```

- The `query()` method of PDO objects can be used to execute an SQL query

PDO

- The `exec()` method of PDO objects can be used to execute an SQL statement returning the number of rows affected by the statement

```
$rowNum = $pdo->exec($statement)  
$rowNum = $pdo->exec("DELETE * FROM meetings")
```

Processing Result Sets

- To get a single row as an array from a result set stored in a PDOStatement object, we can use the `fetch()` method
- By default, PDO returns each row as an array indexed by the `column name` and 0-indexed `column position` in the row

```
$row = $result->fetch()  
array(
```

<https://eduassistpro.github.io>

```
    1 => 'Michael North',  
    2 => 'M.North@student.liverpool.ac.uk')
```

- After the last call of `fetch()` the result se

```
$rows = $result->closeCursor()
```

- To get all rows as an array of arrays from a result set stored in a PDOStatement object, we can use the `fetchAll()` method

```
$rows = $result->fetchAll()
```

Processing Result Sets

- We can use a while-loop together with the `fetch()` method to iterate over all rows in a result set

```
while ($row = $result->fetch()) {  
    echo "Slot: " , $row["slot"] , "<br>\n";  
    echo "Name: " , $row["name"] , " <br>\n";  
    echo "  
}  
}
```

- Altern

<https://eduassistpro.github.io/>

```
foreach($result as $row) {  
    echo "Slot: " , $row["slot"] , "<br>\n";  
    echo "Name: " , $row["name"] , " <br>\n";  
    echo "Email: " , $row["email"] , "<br><br>\n";  
}
```

Add WeChat edu_assist_pro

Processing Result Sets

- Using `bindColumn()` we can bind a variable a particular column in the result set from a query
 - columns can be specified by `number` (starting with 1!)
 - columns can be specified by `name` (matching case)

- Each column in the result set can be bound to a variable that are defined earlier.
- The binding is done using `bindColumn()`.

<https://eduassistpro.github.io/>

```
$result->bindColumn(1, $slot);
$result->bindColumn(2, $name);
$result->bindColumn('email', $email);
while ($row = $result->fetch(PDO::FETCH_BOUND)) {
    echo "Slot: ", $slot, "<br>\n";
    echo "Name: ", $name, "<br>\n";
    echo "Email: ", $email, "<br><br>\n";
}
```

Add WeChat edu_assist_pro

Prepared Statements

- The use of parameterised prepared statements is preferable over queries
- Prepared statements are parsed, analysed, compiled and optimised only once
- Prepared statements can be executed repeatedly with different arguments
- Argument binding injection
- PDO can emulate prepared statements for a DBM that doesn't support them
- MySQL supports prepared statements natively, but the option should be turned off

```
$pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, FALSE);
```

Prepared Statements: SQL Templates

- An **SQL template** is an SQL query (as a string) possibly containing either
 - named parameters of the form `:name` where `name` is a PHP identifier, or
 - question marks ?

for whi

```
$tpl1 = "se  
$tpl2 = "se
```

<https://eduassistpro.github.io>

- The PDO method `prepare()` turns an **SQL statement** (by asking the DBMS to do so)
 - on success, a `PDOStatement` object is returned
 - on failure, `FALSE` or an error will be returned

```
$stmt1 = $pdo->prepare($tpl1);  
$stmt2 = $pdo->prepare("select * from fruit where col=?");
```

Prepared Statements: Binding

- We can bind the parameters of a PDOStatement object to a value using the `bindValue()` method

- Named parameters are bound by name

- Question mark parameters are bound by position (starting from 1!)

- the d
(to m)

- the v

```
$stmt1->bindValue(':name', 'Ben', PDO::PARAM_STR);  
$email = 'bj1@liv.ac.uk';  
$stmt1->bindValue(':email', $email);  
$stmt2->bindValue(1, 20, PDO::PARAM_INT),
```

Add WeChat edu_assist_p

Prepared Statements: Binding

- We can bind the parameters of a PDOStatement object to a variable using the `bindParam()` method

• Named parameters are bound by name

• Question mark parameters are bound by position (starting from 1!)

- the `d`
(to `m`)
- the `v`
- a `val`

<https://eduassistpro.github.io>

```
$name  = 'Ben';
$stmt1->bindParam(':name', $name, PDO::PARAM_STR);
$stmt1->bindParam(':email', $email);
$email = 'bj1@liv.ac.uk';
$slot  = 20;
$stmt2->bindParam(1, $slot, PDO::PARAM_INT);
```

- It is possible to mix `bindParam()` and `bindValue()`

Prepared Statements: Execution

- Prepared statements are executed using `execute()` method
- Parameters must
 - previously have been bound using `bindValue()` or `bindParam()`, or
 - be given as an array of values to `execute`
 - ~ t
 - ~ a
- `exec`
- On `success`, the `PDOStatement` object stores a `result set` (if appropriate)

```
$stmt1->execute();
$stmt1->execute(array(':name' => 'Eve', ':email' => $email));
$stmt2->execute(array(10));
```

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

Transactions

- There are often situations where a single 'unit of work' requires a sequence of database operations
 - e.g. bookings, transfers

Assignment Project Exam Help

- By default, PDO runs in "auto-commit" mode

~ su

- To exe

- only

- rolled back otherwise,

PDO provides the methods

- `beginTransaction()`
 - `commit()`
 - `rollBack()`

Add WeChat edu_assist_pr

Transactions

To support transactions, PDO provides the methods

`beginTransaction()`

- turns off auto-commit mode; changes to the database are **not committed until `commit()` is called**

- returns

- throws a

`commit()`

- changes

auto-commit mode is turned on

- returns **TRUE on success or FALSE on failure**

- throws an **exception** if no transaction is active

`rollBack()`

- discards **changes** to the database; auto-commit mode is restored
- returns **TRUE on success or FALSE on failure**
- throws an **exception** if no transaction is active

Transactions: Example

```
$pdo = new PDO('mysql:host=...;dbname=...', '...', '...',  
    array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,  
          PDO::ATTR_EMULATE_PREPARES => false));  
$pdo->beginTransaction();  
try{  
    $userId = 1, $paymentAmount = 10.50,  
  
    //Query 1: Attempt to insert a payment record  
    $sql = "INSERT INTO accounts SET balance = ? WHERE id = ?";  
    $stmt = $pdo->prepare($sql);  
    $stmt->execute(array($paymentAmount, $userId));  
  
    //Query 2: Attempt to update the user's account  
    $sql = "UPDATE accounts SET balance = balance + ? WHERE id = ?";  
    $stmt = $pdo->prepare($sql);  
    $stmt->execute(array($paymentAmount, $userId));  
  
    // Commit the transaction  
    $pdo->commit();  
} catch(Exception $e){  
    echo $e->getMessage();  
    //Rollback the transaction  
    $pdo->rollBack();  
}
```

Based on <http://thisinterestsme.com/php-pdo-transaction-example/>

Revision

Read

- Language Reference: Classes and Objects

<http://php.net/manual/en/language.oop5.php>

- The PDO Class

<http://php.net/manual/en/pdo.php>

of M. Ado

2017. h

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 14: JavaScript (Part 1)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

Assignment Project Exam Help

③ JavaScript

Motivation

Overview

Examples

<https://eduassistpro.github.io>

④ Types and Variables

Types

Variables

Typecasting

Comparisons

Add WeChat edu_assist_pro

JavaScript: Motivation

- PHP and Perl both allow us to create dynamic web pages
- In web applications, PHP and Perl code is executed on the web server
(server-side scripting)

Assignment Project Exam Help

- allows to use a website template that is instantiated using data stored in a database
- ‘business logic’ is separated from the presentation layer
- not ideal for interactive web applications:
 too slow to react and too much data needs to be transferred
- operations that refer to the location of the user/client
 for example, displaying the local time

```
echo date('H:i l, j F Y');
```

displays the local time on the server not the local time for the user

JavaScript

- JavaScript is a language for client-side scripting
 - script code is embedded in a web page (as for PHP), but delivered to the client is part of the web page and executed by the user's web browser.
 ~ code is visible to the user/client
 - allo
 - dat
 - a web
 - the user may have disallowed the execution of JavaScript code
 - different JavaScript engines may lead to different results not anticipated by the developer of JavaScript
 - performance relies on the efficiency of the JavaScript code running on the client's computing power (not the server's)
 - operations that refer to the location of the client are easy:

```
document.write("Local time: " + (new Date).toString());
```

JavaScript: History

- originally developed by Brendan Eich at Netscape under the name Mocha
- first shipped together with [Netscape browser](#) in September 1995 under the name LiveScript

- obtai

Netsc
in Dece

<https://eduassistpro.github.io/>

- does not have a particularly close relationship to Java; it mixes aspects of Java with aspects of PHP and Perl and its own peculiarities
- is a dialect of ECMAScript, a scripting language standardised in the ECMA-262 specification and ISO/IEC 16262 standard since June 1997
- other dialects include Microsoft's [JScript](#) and [TypeScript](#) and Adobe's [ActionScript](#)

Websites and Programming Languages

Website	Client-Side	Server-Side	Database
Google	JavaScript	C, C++, Go, Java, Python, PHP	BigTable, MariaDB
Facebook			, MySQL, sandra
YouTube			, MariaDB
Yahoo	JavaScript	PHP	reSQL
Amazon	JavaScript	Java, C++, Perl	Oracle DB
Wikipedia	JavaScript	PHP, Hack	MySQL
Twitter	JavaScript	C++, Java, Scala	MySQL
Bing	JavaScript	ASP.NET	MS SQL Server

Wikipedia Contributors: Programming languages used in most popular websites. Wikipedia, The Free Encyclopedia, 20 October 2017, at 11:28. http://en.wikipedia.org/wiki/Programming_languages_used_in_most_popular_websites [accessed 23 October 2017]

JavaScript: Hello World!

```
1 <html><head><title>Hello World</title></head>
2 <body>
3 <p>Our first JavaScript script</p>
4 <script type="text/javascript">
5   document.write("<p><b>Hello World!</b></p>");
6 </script>
7 <noscri
8 JavaScr
9 </noscri
10 </body>
```

<https://eduassistpro.github.io/>

- **JavaScript code** is enclosed between `<script>` and `</script>`
- Alternative HTML markup that is to be used in case JavaScript is not enabled or supported by the web browser, can be specified between `<noscript>` and `</noscript>`
- File must be stored in a directory accessible by the web server, for example `$HOME/public_html`, and be readable by the web server
- No particular file name extension is required

JavaScript scripts

- JavaScript scripts are embedded into HTML documents and are enclosed between `<script>` and `</script>` tags
- A JavaScript script consists of one or more statements and comments
 - ~ there is no need for a main function (or classes)

- Stat

~ S

- Whi
(Th

<https://eduassistpro.github.io>

- One-line comments start with // and run to the end of the line

- Multi-line comments are enclosed in /* and */

- Comments should precede the code they are referring to

Add WeChat edu_assist_pro

Types

- JavaScript is a loosely typed language — like PHP and Perl
- JavaScript distinguished five main types:

Assignment Project Exam Help

- boolean – booleans
- number – integers and floating-point numbers
- str
- fun
- obj

<https://eduassistpro.github.io>

- Integers, floating-point numbers, and strings do not come from the corresponding Perl scalars, including the single-quoted versus double-quoted strings
- JavaScript distinguishes between these five types including between the three primitive types boolean, number and string

Add WeChat `edu_assist_pro`

Variables

- JavaScript variable names do **not** start with a particular character
- A JavaScript variable name may consist of letters, digits, the \$ symbol, and Underscore, but cannot start with a digit
 - ~ you can still stick to the PHP and Perl 'convention' that (s)
- JavaS

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Variables

- Variables can be declared using one of the following statements:

```
var variable1, variable2, ...
```

```
var variable1 = value, variable2 = value, ...
```

Assignment Project Exam Help

- The second statement also initialises the variables
- Use
(only)
- Use <https://eduassistpro.github.io>
- A variable can be initialised without a declaration by assigning a value to it:
`variable = value`
- Both inside and outside a function definition, initialising an undeclared variable creates a global variable
- Note: A declaration does not specify the type of a variable
only assigning a value of a certain type gives a variable a type

Variables

- In JavaScript, the use of the value of a variable that is neither declared nor initialised will result in a reference error and script execution stops
- A declared but uninitialised variable has the default value undefined and has no specific type
- JavaS
requir
- The val

<https://eduassistpro.github.io>

Type	Default	Type	Default	ult
bool	false	string	'undefined'	

Add WeChat edu_assist_pr

```
myVar1++          // reference error
var myVar2
myVar2++          // myVar2 has value NaN
var myVar3
myVar3 = myVar3 + '!' // myVar3 has value 'undefined!'
```

Assignments

- JavaScript uses the equality sign = for assignments

```
student_id = 200846369;
```

As in PHP and Perl, this is an assignment expression

- The value of an assignment expression is the value assigned

```
b = (a = 0) + 1;
```

- JavaS

<https://eduassistpro.github.io/>

Binary assignment	Equ
<i>var</i> $\dagger =$ <i>expr</i>	<i>var</i>
<i>var</i> $-=$ <i>expr</i>	<i>var</i>
<i>var</i> $\ast\ast=$ <i>expr</i>	<i>var</i>
<i>var</i> $/=$ <i>expr</i>	<i>var</i> = <i>var</i> / <i>expr</i>
<i>var</i> $\%=$ <i>expr</i>	<i>var</i> = <i>var</i> % <i>expr</i>

Note: $\ast\ast=$ is not supported

Constants

- Some JavaScript dialects allow the definition of **constants** using

```
const variable1 = value1, variable2 = value2, ...
```

• defines one or more constants

- constants follow the same scope rules as variables

- However, it is not supported by all browsers:
 - and does not work in Internet Explorer 9 and earlier, nor Opera before version 12

Add WeChat edu_assist_pro

Values, Variables and Types

- string `typeof value`

returns a string representation of the type of *value*

Boolean	Boolean	Number	number
String	"string"	Object	"object"
und			
NaN			
Futur			

<https://eduassistpro.github.io>

`typeof null` to "null" (as in PHP)

```
document.writeln("Type of 23.0: " + typeof(23.0) + "<br />")
document.writeln("Type of \"23\": " + typeof("23") + "<br />")
var a
document.writeln("Type of a: " + typeof(a) + "<br />")
```

```
Type of 23.0: number<br />
Type of "23": string<br />
Type of a: undefined<br />
```

Typecasting

JavaScript provides several ways to explicitly **type cast** a value

- Apply an identity function of the target type to the value

"12" * 1

12 + ""

false

[12,

12

"12"

!!"0"

[12]

"12" * 1

!!"0"

!!"0"

!!"0"

true

true

false

true

NaN

12

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Typecasting

JavaScript provides several ways to explicitly **type cast** a value

- Wrap a value of a primitive type into an object

JavaScript has objects Number, String, and Boolean with unary constructors/wrappers for values of primitive types

(J

Numb

Stri

String

(false)

\rightsquigarrow "false"

Number

(true)

1

true
true

<https://eduassistpro.github.io>

- Use **parser functions** parseInt or parseFloat

parseInt("12")

\rightsquigarrow 12

par

2.5

parseInt("2.5")

\rightsquigarrow 2

parseFloat("2.5e1")

\rightsquigarrow 25

parseInt("E52")

\rightsquigarrow NaN

parseFloat("E5.2")

\rightsquigarrow NaN

parseInt("42")

\rightsquigarrow 42

parseFloat("4.2")

\rightsquigarrow 4.2

parseInt("2014Mar")

\rightsquigarrow 2014

parseFloat("4.2end")

\rightsquigarrow 4.2

Comparison operators

JavaScript distinguishes between (loose) equality ==
and strict equality === in the same way as PHP:

$expr1 == expr2$	Equal	TRUE iff $expr1$ is equal to $expr2$ after type coercion
$expr1 != expr2$	Not equal	TRUE iff $expr1$ is not equal to $expr2$

- When comparing with a `number`, the `number` is converted to a `string` and to `0` if `false`
- When comparing with a `boolean`, the `boolean` is converted to `true` and to `0` if `false`
- If an `object` is compared with a `number` or `string`, JavaScript uses the `valueOf` and `toString` methods of the objects to produce a primitive value for the object
- If two `objects` are compared, then the equality test is true only if both refer to the same object

Comparison operators

JavaScript distinguishes between (loose) equality ==
and strict equality === in the same way as PHP:

$expr1 === expr2$	Strictly equal	TRUE iff $expr1$ is equal to $expr2$, and they are of the same type
$expr1 !== expr2$	Strictly not	TRUE iff $expr1$ is not equal to $expr2$,

"123" == 123	~	false
"123" != 123	~	true
"1.23e2" == 123	~	true
"1.23e2" == "12.3e1"	~	false
5 == true	~	false

Add WeChat edu_assist_pro

Comparison operators

JavaScript's comparison operators also applies **type coercion** to their operands and do so following the same rules as equality ==:

$expr1 < expr2$ Less than true iff $expr1$ is strictly less than $expr2$ after type coercion

$expr1 > expr2$ Greater than

true iff $expr1$ is strictly greater than $expr2$

$expr1$

$expr1$

or equal to

'35.5' > 35

'ABD' > 'ABC'

'1.23e2' > '12.3e1'

"F1" < "G0"

true > false

5 > true

~ true

~ true

~ false

~ true

~ true

~ true

afte

'35' > '35' true

'AB' > 'AB' true

'1.23e2' > '12.3e1' ~ false

"F1" < "G0" ~ true

true >= false ~ true

5 >= true ~ true

$https://eduassistpro.github.io$

Assignment Project Exam Help

Equality

Why do we care whether `5 == true` is true or false?

~ it influences how our scripts behave

~ it influences whether more complex objects are equal or not

Assignment Project Exam Help

PHP:

```
if (5) print(" i  
else print("5 is not t  
print(" and ")  
if (5 == true) prin  
            else print("5 is not equal to true");
```

Output: 5 is true and 5 is equal to true

JavaScript:

```
if (5) document.writeln("5 is true");  
    else document.writeln("5 is not true")  
document.writeln(" and ")  
if (5 == true) document.writeln("5 is equal to true")  
            else document.writeln("5 is not equal to true")
```

Output: 5 is true and 5 is not equal to true

Equality

Why do we care whether `5 == true` is true or false?

~ it influences how our scripts behave

~ it influences whether more complex objects are equal or not

Assignment Project Exam Help

PHP:

```
$array3 = arra
$array4 = arra
if (($array3
     pr
else print("The two arrays are not equal");
```

Output: The two arrays are equal

Add WeChat edu_assist_pr

JavaScript:

```
$array3 = ["1.23e2",5]
$array4 = ["12.3e1",true]
if (($array3[1] == $array4[1]) && ($array3[2] == $array4[2]))
    document.writeln("The two arrays are equal")
else document.writeln("The two arrays are not equal")
```

Output: The two arrays are not equal

Equality

Note: The way in which more complex data structures are compared also differs between PHP and JavaScript

Assignment Project Exam Help

PHP:

```
$array3 = array("1.23e2",5);
$array4 = arra
if ($array3 == $a
    pr
else print("
```

<https://eduassistpro.github.io>

Output: The two arrays are equal

Add WeChat edu_assist_pro

```
$array3 = ["1.23e2",5]
$array5 = ["1.23e2",5]
if ($array3 == $array5)
    document.writeln("The two arrays are equal")
else document.writeln("The two arrays are not equal")
```

Output: The two arrays are not equal

Revision

Assignment Project Exam Help

Read
• Chapter 14: Exploring JavaScript
of
R. Nixon:
Learnin
O'Reilly, 2009.

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 15: JavaScript (Part 2)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

- 35 Primitive datatypes

Numbers
Booleans
Strings

- 36 Array

De
for

Array functions

- 37 Control structures

Conditional statements

Switch statements

While- and Do While-loops

For-loops

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Integers and Floating-point numbers

- The JavaScript datatype `number` covers both

- integer numbers

0

2012

-40

1263978

- Floating-point numbers

1.23

256.0

-12e19

2.4e-10

Assignment Project Exam Help

- The `Math` object provides a wide range of mathematical functions

`Math`

`Math`

`Math`

`Math`

`Math.log(number)`

natural logari

`Math.random()`

random num

`Math.sqrt(number)`

square root

Add WeChat `edu_assist_pro`

- There are also some pre-defined number constants including

`Math.PI` (case sensitive) 3.14159265358979323846

`NaN` (case sensitive) 'not a number'

`Infinity` (case sensitive) 'infinity'

Numbers: NaN and Infinity

- The constants `NaN` and `Infinity` are used as `return values` for applications of mathematical functions that do not return a number

- `Math.log(0)` returns `-Infinity` (negative 'infinity')
- `Math.sqrt(-1)` returns `NaN` ('not a number')

- `1/0`
- `0/0`

<https://eduassistpro.github.io>

- Equality and comparison operators produce the following results for `NaN` and `Infinity`:

<code>NaN == NaN</code>	\rightsquigarrow false
<code>Infinity == Infinity</code>	\rightsquigarrow true
<code>NaN == 1</code>	\rightsquigarrow false
<code>NaN < NaN</code>	\rightsquigarrow false
<code>1 < Infinity</code>	\rightsquigarrow true
<code>Infinity < 1</code>	\rightsquigarrow false
<code>NaN < Infinity</code>	\rightsquigarrow false

<code>NaN == -NaN</code>	\rightsquigarrow false
<code>Infinity < -Infinity</code>	\rightsquigarrow false
<code>Infinity == 1</code>	\rightsquigarrow false
<code>Infinity < Infinity</code>	\rightsquigarrow false
<code>1 < NaN</code>	\rightsquigarrow false
<code>NaN < 1</code>	\rightsquigarrow false
<code>Infinity < NaN</code>	\rightsquigarrow false

Integers and Floating-point numbers: NaN and Infinity

- JavaScript provides two functions to test whether a value is or is not NaN, Infinity or -Infinity:

- `bool isNaN(value)`
returns TRUE iff *value* is NaN

- `bool isFinite(value)`
retu

There

ty
<https://eduassistpro.github.io>

- In conversion to a boolean value,

- NaN converts to `false`
- Infinity converts to `true`

- In conversion to a string,

- NaN converts to '`NaN`'
- Infinity converts to '`Infinity`'

Booleans

- JavaScript has a **boolean** datatype with constants **true** and **false** (case sensitive)
- JavaScript offers the same short circuit boolean operators as Java, Perl and PHP:

But a <https://eduassistpro.github.io/>

- The **truth tables** for these operators are the same as for taking into account that the conversion of non-boolean values differs
- Remember that **&&** and **||** are **not** commutative, that is,
 $(A \&\& B)$ is not the same as $(B \&\& A)$
 $(A || B)$ is not the same as $(B || A)$

Type conversion to boolean

When `converting to boolean`, the following values are considered `false`:

- the boolean `false` itself
- the number `0` (zero)
- the empty string, **but not the string '0'**
- `undefined`
- `null`
- `Nan`

<https://eduassistpro.github.io>

Every other value is converted to `true` in

- `Infinity`
- `'0'`
- functions
- objects, in particular, `arrays with zero elements`

Strings

- JavaScript supports both **single-quoted** and **double-quoted** strings
- JavaScript uses **+** for **string concatenation**
- Within **double-quoted strings**, JavaScript supports the following escape characters

\		wline)
\		(backslash)
\		

- JavaScript does **not** support variable interpolation
- JavaScript also does **not** support **heredocs**, but multi-line strings are possible

```
document.writeln("Your \
    name is " + name + "and\
    you are studying " + degree + "\\
    at " + university);
```

Arrays

- An array is created by assigning an array value to a variable

```
var arrayVar = []
var arrayVar = [element0, element1, ...]
```

- JavaScript uses

arrayV

to denote
The first

<https://eduassistpro.github.io/>

- Arrays have no fixed length and it is always possible to add elements to an array
- Accessing an element of an array that has not been assigned returns undefined
- For an array arrayVar, arrayVar.length returns the maximal index index such that arrayVar[index] has been assigned a value (including the value undefined) plus one

Add WeChat edu_assist_pro

Arrays

- It is possible to assign a value to `arrayVar.length`
 - if the assigned value is greater than the previous value of `arrayVar.length`, then the array is ‘extended’ by additional `undefined` elements
 - if the assigned value is smaller than the previous value of `arrayVar.length`, then array elements with greater or equal index will be deleted

- Assign origin <https://eduassistpro.github.io>
 - ~ changes to the new variable affect the original array
- Arrays are also passed to functions by reference
- The `slice` function can be used to create a proper object `arrayVar.slice(start, end)`
returns a copy of those elements of array `variable` that have indices between `start` and `end`

Arrays: Example

```
var array1 = ['hello', [1, 2], function() {return 5}, 43]
document.writeln("1: " + array1.length + array1.length + "<br>")
1: array1.length = 4<br>
document.writeln("2: " + array1[3] + array1[3] + "<br>")
2: array1[3] = 43<br>
array1[
document
3: array
document
4: array1[4] = undefined<br>
document.writeln("5: " + array1[5] + array1[
5: array1[5] = world<br>
array1.length = 4
document.writeln("6: " + array1[5] + array1[
6: array1[5] = undefined<br>
var array2 = array1
array2[3] = 7
document.writeln("7: " + array1[3] + array1[3] + "<br>")
7: array1[3] = 7<br>
```

Add WeChat edu_assist_pr

forEach-method

- The recommended way to iterate over all elements of an `array` is a `for-loop`

```
for (index = 0; index < arrayVar.length; index++) {  
    arrayVar[index]  
}
```

- An alte

```
var c  
s  
}  
array.forEach(callback);
```

- The `forEach` method takes a function as an argument
- It iterates over all indices/elements of an array
- It passes the current array element (`elem`), the current index (`index`) and a pointer to the array (`arrayArg`) to the function
- Return values of that function are ignored, but the function may have `side effecs`

forEach-method: Example

```
var myArray = ['Michele Zito', 'Ullrich Hustadt'];
```

```
var rewriteNames = function (elem, index, arr) {
    arr[index] = elem.replace(/\b(\w+)\b\s+(\w+)/, "$2,$1");
}
```

```
myArray.f
for (i=0; i<my
    document.write('['+i+']= '+myArray[i]+")";
}
document.writeln('<br>');
```

[0] = Zito, Michele [1] = Hustadt, Ullrich

Array operators

JavaScript has no `stack` or `queue` data structures, but has `stack` and `queue` functions for arrays:

- `number array.push(value, value, ...)`

appends one or more elements at the end of an array;
return

- `mixed array.pop()`
extract

- `mixed array.shift()`

shift extracts the first element of an array and returns it

- `number array.unshift(value, va`

inserts one or more elements at the start of an array variable;
returns the number of elements in the resulting array

Note: In contrast to PHP and Perl, `array` does **not** need to be a `variable`

Array operators: push, pop, shift, unshift

```
planets = ["earth"]
planets.unshift("mercury", "venus")
planets.push("mars", "jupiter", "saturn");
document.writeln("planets@1:" + planets.join(" ") + "<br>")
planets@1: mercury venus earth mars jupiter saturn <br>
last = planets.pop()
document
planets@2:
first = planets[0]
document.writeln("planets@3:" + planets.join(" ") + "<br>")
planets@3: venus earth mars jupiter <br>
document.writeln("planets@4:" + first + "<br>@"
    @4: mercury saturn <br>
home = ["mercury", "venus", "earth"].pop()
document.writeln("planets@5:" + home + "<br>")
    @5: earth <br>
number = ["earth"].push("mars");
document.writeln("planets@6:" + number + "<br>")
    @6: 2 <br>
```

Control structures

Assignment Project Exam Help

JavaScript control structures

- condit
- switc
- while-
- for-loops
- break and continue

Add WeChat edu_assist_pr

are identical to those of PHP except for conditional stat

Control structures: conditional statements

JavaScript **conditional statements** do not allow for **elsif**- or **elseif**-clauses, but conditional statements can be nested:

```
if (condition) {  
    statements  
} else if (condition) {  
    sta  
} else {  
    sta  
}
```

<https://eduassistpro.github.io>

- The **else-clause** is optional but there can be at most one
- Curly Brackets can be omitted if there is only a single statement in a clause

JavaScript also supports **conditional expressions**

```
condition ? if_true_expr : if_false_expr
```

Control structures: switch statement

Switch statements in JavaScript take the same form as in PHP:

```
switch (expr) {  
    case expr1:  
        statements  
    case expr2:  
        statements  
    ...  
    default:  
        statements  
}
```

- there can be arbitrarily many `case`-clauses
- the `default`-clause is optional but there can be at most one

`case`

<https://eduassistpro.github.io/>

`default:`

`statements`

corresponding clause

`break;`

if none of `expr1`

then the `default`-clause

Add WeChat `edu_assist_pro`

- `break` ‘breaks out’ of the switch statement
- if a clause does not contain a `break` command, then execution moves to the next clause

Control structures: switch statement

Not every `case`-clause needs to have associated statements

Example:

```
switch (month) {  
    case 1:    case 3:    case 5:    case 7:  
    case 8:    case 10:   case 12:  
        da  
        br  
    case 4:    case 6:    case 9:    case 11:  
        days = 30;  
        break;  
    case 2:  
        days = 28;  
        break;  
    default:  
        days = 0;  
        break;  
}
```

Add WeChat edu_assist_p

Control structures: while- and do while-loops

- JavaScript offers `while-loops` and `do while-loops`

```
while (condition) {  
    statement  
}
```

Assignment Project Exam Help

```
do {  
    st  
} while (condition)
```

<https://eduassistpro.github.io/>

- As usual, `curly brackets` can be omitted if the loop consists of only one statement

Example: Add WeChat `edu_assist_pro`

```
// Compute the factorial of a given number  
factorial = 1;  
do {  
    factorial *= number--;  
} while (number > 0);
```

Control structures: for-loops

- for-loops in JavaScript take the form

```
for (initialisation; test; increment) {  
    statement  
}
```

Assignment Project Exam Help

- Again
- consists
- In Java, can consist of more than one statement, separated by commas instead of semicolons

Example:

```
for (i = 3; j = 3; j >= 0; i++, j--)  
    document.writeln(i + " - " + j + " - " + i*j)  
    // Indentation has no 'meaning' in JavaScript,  
    // the next line is not part of the loop  
    document.writeln("After loop: " + i + " - " + j)
```

- Note: Variables introduced in a for-loop are still global even if declared using var

Control structures: break and continue

- The `break` command can also be used in while-, do while-, and for-loops and discontinues the execution of the loop

```
while (value < 100) {  
    if (value == 0) break;  
    value++  
}
```

- The `continue` command loops an

```
for (x = -2; x <= 2; x++) {  
    if (x == 0) continue;  
    document.writeln("10 / " + x + " = " + 10 / x);  
}
```

10 / -2 = -5
10 / -1 = -10
10 / 1 = 10
10 / 2 = 5

Revision

Assignment Project Exam Help

Read

- Chapter 15: Expressions and Control Flow in JavaScript
- Chapter 16: Functions

of

R. Nixon:
Learning PHP, MySQL, and JavaScript.
O'Reilly, 2009

Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 16: JavaScript (Part 3)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

38 Functions

Defining a function

Calling a function

Variable-length argument lists

Static variables

Exa

Nes

39 JavaS

40 (User-defined) Objects

Object Literals

Object Constructors

Definition and use

Prototype property

Public and private static variables

Pre-defined objects

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Functions

Function definitions can take several different forms in JavaScript including:

```
function identifier(param1, param2, ...) {  
    statements  
}
```

Assignment Project Exam Help

```
var ide  
    statem
```

- Such functions can be defined in the body of a HTML page or in a library that is then imported
- Function names are case-sensitive
- The function name must be followed by parentheses
- A function has zero, one, or more parameters that are variables
- Parameters are not typed
- identifier.length* can be used inside the body of the function to determine the number of parameters

Add WeChat edu_assist_pro

Functions

Function definitions can take several different forms in JavaScript including:

```
function identifier(param1, param2, ...) {  
    statements  
}
```

```
var ide  
    statem
```

- The `ret`

```
        return value
```

can be used to terminate the execution of a function an

`value` the return value of the function

- The `return value` does **not** have to be of a primitive type
- A function can contain more than one return statement
- Different return statements can return values of different types
 - ~ there is no `return type` for a function

Calling a function

A function is **called** by using the function name followed by a list of **arguments** in parentheses

```
function identifier(param1, param2, ...) {  
    ...  
}  
... ide
```

- The **list** of parameters must have the same number of elements as the **list** of arguments.
- If it is shorter, then any parameter without corresponding argument will have value **undefined**.

Add WeChat edu_assist_pro

```
function sum(num1, num2) { return num1 + num2 }

sum1 = sum(5, 4)          // sum1 = 9
sum2 = sum(5, 4, 3)       // sum2 = 9
sum3 = sum(5)             // sum3 = NaN
```

'Default values' for parameters

- JavaScript does **not** allow to specify **default values** for function parameters
- Instead a function has to check whether a parameter has the value **undefined** and take appropriate action

```
function sum  
  if (num1 = u)  
  if (num2 = u)  
    return num1 + num2  
}  
  
sum3 = sum(5) // sum3 = 5  
sum4 = sum()   // sum4 = 0
```

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Variable-length argument lists

- Every JavaScript function has a property called `arguments`
- The `arguments` property consists of an array of all the arguments passed to a function
- As for any JavaScript array, `arguments.length` can be used to determine

```
function sum()
  if (arguments.length > 0)
    sum = 0
    for (var i=0; i<arguments.length; i++)
      sum = sum + arguments[i]
    return sum
}
```

```
sum0 = sumAll()           // sum0 = null
sum1 = sumAll(5)          // sum1 = 5
sum2 = sumAll(5,4)         // sum2 = 9
sum3 = sumAll(5,4,3)       // sum3 = 12
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

JavaScript functions and Static variables

- JavaScript does not have a `static` keyword to declare a variable to be static and preserve its value between different calls of a function
- The solution is to use a `function property` instead

Assignment Project Exam Help

```
function counter() {  
    counte  
    counte  
    return co  
}  
  
document.writeln("1: static count = "+counter())  
document.writeln("2: static count = "+counter())  
document.writeln("3: global counter.count = "+counter().count)  
  
1: static count = 1  
2: static count = 2  
3: global counter.count = 2
```

- As the example shows the `function property` is global/public
- `Private static variables` require more coding effort

JavaScript functions: Example

```
function bubble_sort(array) {  
    if (!(array && array.constructor == Array))  
        throw("Argument not an array")  
    for (var i=0; i<array.length; i++) {  
        for (var j=0; j<array.length-i; j++) {  
            } } }  
    return array  
}  
  
function swap(array, i, j) {  
    var tmp = array[i]  
    array[i] = array[j]  
    array[j] = tmp  
}
```

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

JavaScript functions: Example

```
function bubble_sort(array) { ... }  
function swap(array, i, j) { ... }
```

```
array = [2, 4, 3, 9, 6, 8, 5, 1]  
document.writeln("array before sorting "+  
    array.join(", ")+"  
    <br>")
```

```
array before sorting 2, 4, 3, 9, 6, 8, 5, 1 <br>
```

```
sorted = bubble_
```

```
sorted = bubble_sort(array)  
document.writeln("array after sorting of itself "+  
    array.join(", ")+"  
    <br>")
```

```
array after sorting of copy 2, 4, 3, 9, 6, 8, 5, 1 <br>
```

```
sorted = bubble_sort(array)  
document.writeln("array after sorting of itself "+  
    array.join(", ")+"  
    <br>")
```

```
array after sorting of itself 1, 2, 3, 4, 5, 6, 8, 9 <br>
```

```
document.writeln("sorted array "+  
    sorted.join(", ")+"  
    <br>")
```

```
sorted array 1, 2, 3, 4, 5, 6, 8, 9 <br>
```

Nested function definitions

- Function definitions can be nested in JavaScript
- Inner functions have access to the variables of outer functions
 - By default, inner functions can not be invoked from outside the function they are defined in

```
function bub
  function swap(i, j)
    // swap(i, j) -> swap(j, i)
    // a local variable of the outer function bubble_sort
    var tmp = array[i]; array[i] = array[j]; array[j] = tmp;
  }
  if (!array || !array.constructor === Array)
    throw("Argument must be an array")
  for (var i=0; i<array.length; i++) {
    for (var j=0; j<array.length-i; j++) {
      if (array[j+1] < array[j]) swap(j, j+1)
    }
  }
  return array }
```

JavaScript libraries

- Collections of JavaScript functions (and other code), **libraries**, can be stored in one or more files and then be reused
- By convention, files containing a **JavaScript library** are given the file name extension **.js**
- <scr
- A Java
`<script type="text/javascript"
src="http://cgi.csc.liv.ac.uk/~`

https://eduassistpro.github.io

where **url** is the (relative or absolute) URL for lib

```
<script type="text/javascript"  
src="http://cgi.csc.liv.ac.uk/~
```

Add WeChat edu_assist_pro

- One such import statement is required for each library
- Import statements are typically placed in the **head section** of a page or at the end of the **body section**
- Web browsers typically cache libraries

JavaScript libraries: Example

```
~ullrich/public_html/sort.js
```

```
function bubble_sort(array) {  
    ... swap(array, i, j+1); ...  
    return array;  
}
```

```
function swap
```

```
example
```

```
<html><head><title>Sorting example</title>
```

```
<script type="text/javascript"  
src="http://csci.cscliv.ac.uk/~ul  
</script></head>
```

```
<body>
```

```
<script type="text/javascript">  
array = [2,4,3,9,6,8,5,1];  
sorted = bubble_sort(array.slice(0))  
</script>  
</body></html>
```

Assignment Project Exam Help
<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Object Literals

- JavaScript is an object-oriented language, but one without **classes**
- Instead of defining a class,

We can simply state an object literal:

```
{ property1: value1, property2: value2, ... }
```

where

and

<https://eduassistpro.github.io>

```
var person1 = {  
    age: (30 + 2),  
    gender: 'male',  
    name: { first: 'Bob', last: 'Smith' },  
    interests: ['music', 'skiing'],  
    hello: function() { return 'Hi! I\'m ' + this.name.first + '.' }  
};
```

```
person1.age          --> 32           // dot notation  
person1['gender']   --> 'male'        // bracket notation  
person1.name.first  --> 'Bob'  
person1['name']['last'] --> 'Smith'
```

Object Literals

```
var person1 = {  
    ...  
    name: { first: 'Bob', last: 'Smith' },  
    hello: function() { return 'Hi! I'm ' + this.name.first + '.' }  
};  
person1.hello() --> "Hi! I'm Bob."
```

- Every p
execu g to
- Every e
itself
- In person1.hello(), the execution context
~ this.name.first is person1.name

Add WeChat edu_assist_pro

Object Literals

```
var person1 = {  
    name: { first : 'Bob', last : 'Smith' },  
    greet: function() { return 'Hi! I\'m ' + name.first + '.' },  
    full1: this.name.first + " " + this.name.last,  
    full2: name.first + " " + name.last  
};
```

```
person1.gr      -  
person1.fu      --> "unde  
person1.full1  --> "unde
```

- In `person1.greet`, `this` refers to `person1`
 ~ but `name.first` does **not** refer to `p`
- In the (construction of the) object literal itself,
 `person1` but its **execution context** (the global environment)
 ~ none of `name.first`, `name.last`, `this.name.first`, and
 `this.name.last` refers to properties of this object literal

Add WeChat `edu_assist_pro` to your contacts.

Objects Constructors

- JavaScript is an object-oriented language, but one without **classes**

- Instead of defining a class,
we can define a **function** that acts as **object constructor**:

- variables declared inside the function will be **instance variables** of the object
 - ~ e

- it is po

- inne

<https://eduassistpro.github.io>

- it is possible to make such functions/methods **privat**

- private variables/methods can only be accessed ins

- public variables/methods can be accessed outside

Add WeChat edu_assist_pro

- Whenever an **object constructor** is called,
prefixed with the keyword **new**, then

- a new object is created

- the function is executed with the keyword **this** bound to that object

Objects: Definition and use

```
function SomeObj() {  
    instVar2      = 'B'          // private variable  
    var instVar3  = 'C'          // private variable  
  
    this.instVar1 = 'A'          // public variable  
    this.method1 = function() { // public method  
        // use of a public variable, e.g. 'instVar1', must be preceded by 'this'  
        return 'm'  
  
    this.method2 = function() { // public method  
        // can't just do public  
        return '123' +  
    }  
  
    method3 = function() {      // private method  
        return 'm3[' + instVar2 + ']' + method4()  
    }  
  
    var method4 = function() { // private method  
        return 'm4[' + instVar3 + ']'  
    }  
}  
obj = new SomeObj()                      // creates a new object
```

```
obj.instVar1      --> "A"  
obj.instVar2      --> undefined  
obj.instVar3      --> undefined  
obj.method1()     --> "m1[A] m3[B] m4[C]"  
obj.method2()     --> "m2[m1[A] m3[B] m4[C]]"  
obj.method3()     --> error  
obj.method4()     --> error
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Objects: Definition and use

```
function SomeObj() {  
    this.instVar1 = 'A'           // public variable  
    this.instVar2 = 'B'           // private variable  
    var instVar3 = 'C'           // private variable  
  
    this.m  
    this.m  
    method1 = f  
    var method4 = function() { ... }  
}
```

Add WeChat edu_assist_pro

- Note that all of instVar1 to instVar3 are **instance variables (properties, members)** of someObj
- The only difference is that instVar1 to instVar3 store strings while method1 to method4 store functions
 - ~ every object stores its own copy of the methods

Objects: Prototype property

- All functions have a `prototype` property that can hold shared object properties and methods

objects do not store their own copies of these properties and methods but only store references to a single copy

```
function Som
  this.j
  instVa      = 'B'
  var instVar3 = 'C'          // private va
  SomeObj.prototype.method1 = function() { ... }
  SomeObj.prototype.method2 = function() { ... }

  method3 = function() { ... }      // private method
  var method4 = function() { ... }  // private method
}
```

Add WeChat edu_assist_pro

Note: `prototype` properties and methods are always **public!**

Objects: Prototype property

- The `prototype` property can be modified 'on-the-fly'
 - all already existing objects gain new properties / methods
 - manipulation of properties / methods associated with the `prototype` property needs to be done with care

```
function Some
obj1 = new SomeObj
obj2 = new SomeObj
document.writeln(obj1.instVar4) // 'A'
document.writeln(obj2.instVar4) // 'A'

SomeObj.prototype.instVar4 = 'B'
document.writeln(obj1.instVar4) // 'B'
document.writeln(obj2.instVar4) // 'B'

obj1.instVar4 = 'C' // creates a new instance variable for obj1
SomeObj.prototype.instVar4 = 'D'
document.writeln(obj1.instVar4) // 'C' !!
document.writeln(obj2.instVar4) // 'D' !!
```

Add WeChat edu_assist_pro

Objects: Prototype property

- The `prototype` property can be modified 'on-the-fly'
 - all already existing objects gain new properties / methods
 - manipulation of properties / methods associated with the `prototype` property needs to be done with care

```
function SomeObj
obj1 = new SomeObj
obj2 = new SomeObj
SomeObj.prototype.instVar5 = 'E'

SomeObj.prototype.setInstVar5 = function(arg) {
  this.instVar5 = arg
}

obj1.setInstVar5('E')
obj2.setInstVar5('F')

document.writeln(obj1.instVar5) // 'E' !!
document.writeln(obj2.instVar5) // 'F' !!
```

Add WeChat edu_assist_pro

'Class' variables and 'Class' methods

Function properties can be used to emulate Java's `class variables` (static variables shared among instances) and `class methods`

```
function Circle(radius) { this = radius; }
// 'class variable' - property of the Circle constructor function
Circle.PI = 3.14159
```

```
// 'instance meth
Circle.prototype.area = function () {
    return this.PI * this.r * this.r;
}
```

```
// 'class method' - property of the Circle constructor function
Circle.max = function (cx,cy) {
    if (cx > cy) { return cx; } else { return cy; }
}
```

```
c1      = new Circle(1.0)      // create an instance of the Circle class
c1.r    = 2.2;                  // set the r instance variable
c1_area = c1.area();           // invoke the area() instance method
x      = Math.exp(Circle.PI)   // use the PI class variable in a computation
c2      = new Circle(1.2)       // create another Circle instance
bigger  = Circle.max(c1,c2)    // use the max() class method
```

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat `edu_assist_pro`

Private static variables

In order to create **private static variables** shared between objects we can use a **self-executing anonymous function**

```
var Person = (function () {  
    var population = 0; // private static 'class' variable
```

```
        return function (value) { // constructor  
            popula  
            var name = value  
            this.s  
            this.g  
            this.g  
        }  
    }())
```

```
person1 = new Person('Peter')  
person2 = new Person('James')
```

```
person1.getName() --> 'Peter'  
person2.getName() --> 'James'  
person1.name --> undefined  
Person.population || person1.population --> undefined  
person1.getPop() --> 2  
person1.setName('David')  
person1.getName() --> 'David'
```

Pre-defined objects: String

- JavaScript has a collection of pre-defined objects, including `Array`, `String`, `Date`

A `String` object encapsulates values of the primitive datatype `string`.

- Properties of a `String` object include

- `len`

- Meth

- `ch`

the character at position `index` (counting from 0)

- `substring(start, end)`

returns the part of a string between positions `start` and `end` (exclusive)

- `toUpperCase()`

returns a copy of a string with all letters in uppercase

- `toLowerCase()`

returns a copy of a string with all letters in lowercase

Pre-defined objects: String and RegExp

- JavaScript supports (Perl-like) **regular expressions** and the **String** objects have methods that use regular expressions:

Assignment Project Exam Help

- **search(*regexp*)**
matches *regexp* with a string and returns the start position of the first match if found, -1 if not
- **match(*regexp*)**
– with *i* or *I*
– with *g* modifier returns an array containing all the matches in the whole expression
- **replace(*regexp*, *replacement*)**
replaces matches for *regexp* with *replacement* and returns the resulting string

```
name1 = 'Dave Shield'.replace(/(\w+)\s(\w+)/, "$2, $1")
regexp = new RegExp("(\\w+)\\s(\\w+)")
name2 = 'Ken Chan'.replace(regexp, "$2, $1")
```

Pre-defined objects: Date

- The `Date` object can be used to access the (local) date and time
- The `Date` object supports various `constructors`

`new Date()` current date and time

`new Date(milliseconds)` set date to milliseconds since 1 Januar 1970

`new D`

^{ng}

<https://eduassistpro.github.io>

- Methods provided by `Date` include

`toString()`

returns a string representation of the `Date` object

`getFullYear()`

returns a four digit string representation of the (current) year

`parse()`

parses a date string and returns the number of milliseconds since midnight of 1 January 1970

Revision

Read

- Chapter 16: JavaScript Functions, Objects, and Arrays
- Chapter 17: JavaScript and PHP Validation and Error Handling
(Regular Expressions)

of

R. Nixon:

Learnin

<https://eduassistpro.github.io>

O'Reilly, 2009.

- <http://coffeeonthekboard.com/private-variables-in-javascript>
- <http://coffeeonthekboard.com/javascript-private-static-members-part-1-208/>
- <http://coffeeonthekboard.com/javascript-private-static-members-part-2-218/>

Assignment Project Exam Help

COMP284 Scripting Languages
Lecture 17: JavaScript (Part 4)
Handouts

<https://eduassistpro.github.io>

Department of Computer
School of Electrical Engineering, Electronics, and Computing
University of Liverpool

Add WeChat edu_assist_pro

Contents

Assignment Project Exam Help

- ④ Dynamic web pages using JavaScript
Window and Document objects

Wi

Dia

Inp

Do

<https://eduassistpro.github.io>

- ② Event-driven Programs
 - Introduction
 - Events

Window and Document objects

JavaScript provides two objects that are essential to the creation of dynamic web pages and interactive web applications:

Assignment Project Exam Help

- a JavaScript object that represents a browser window or tab

- auto

- <fra

- <https://eduassistpro.github.io>

- allows

- ~> JavaScript provides methods that allow windows created and manipulated.

- Example: `window.open('http://www.c')`

- whenever an object method or property is referenced in a script without an object name and dot prefix it is assumed by JavaScript to be a member of the `window object`

- Example: We can write `alert()` instead of `window.alert()`

Window object

- A **window object** represents an open window in a browser.
- If a document contain **frames**, then there is
 - one **window object**, `window`, for the HTML document
 - and one additional window object for each frame,
accessed via the `name` property of the frame's `document` object.
- A **window object** has the following properties:

<https://eduassistpro.github.io/>

doc	document object for the window
history	history object for the window
location	location object (current URL)
navigator	navigator (web browser) object
opener	reference to the window that opened this window
innerHeight	inner height of a window's content area
innerWidth	inner width of a window's content area
closed	boolean value indicating whether the window is (still) open

Navigator object

Properties of a `navigator object` include

<code>navigator.appName</code>	the web browser's name
<code>navigator.appVersion</code>	the web browser's version

Assignment Project Exam Help

Example: Load different style sheets depending on browser

```
<html><head>
<script type="text/javascript">
if (navigator.appName == "Netscape") {
    document.writeln('<link rel="stylesheet" type="text/css" ' +
                    'href="Netscape.css">')
} else if (navigator.appName == "Opera") {
    document.writeln('<link rel="stylesheet" type="text/css" ' +
                    'href="Opera.css">')
} else {
    document.writeln('<link rel="stylesheet" type="text/css" ' +
                    'href="Others.css">')
}
</script></head>
```

Window object

Methods provided by a window object include

- `open(url, name [, features])`
- opens a new browser window/an
- returns a reference to a window object
- `url`
- `nam`
- `fea`

Assignment Project Exam Help

<https://eduassistpro.github.io>

The standard sequence for the creation of a new window

```
// new instance of 'Window' class  
var newWin = new Window(...)  
newWin.document.write('<html>...</html>')
```

instead it is

```
// new window created by using 'open' with an existing one  
var newWin = window.open(...)  
newWin.document.write('<html>...</html>')
```

Window object

Methods provided by a window object include

- `close()`
 - Closes a browser window/tab
- `focus()`
 - give f
- `blur`
 - rem
- `print()`
 - prints (sends to a printer) the contents of the current wi

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

Window object: Example

```
<html><head><title>Window handling </title>
<script type="text/javascript">
function Help() {
    var OutputWindow = window.open(' ', 'Help', 'resizable=1');
    with (OutputWindow.document) {
        open()
        wri
        </h
        mes
        pag
        close()
    }
}
</script></head><body>
<form name="ButtonForm" id="ButtonForm" action="">
<p>
    <input type="button" value="Click for Help"
           onclick="Help(); ">
</p>
</form></body></html>
```

Assignment Project Exam Help
<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Window object: Dialog boxes

- Often we only want to open a new window in order to
 - display a message
 - ask for confirmation of an action
 - request an input

Assignment Project Exam Help

- For the pre-defined <https://eduassistpro.github.io/windows> (windows for simple dialogs):
 - `null alert(message_string)`
 - `bool confirm(message_string)`
 - `string prompt(message_string, default)`

Add WeChat edu_assist_pro

Window object: Dialog boxes

- `null alert(message_string)`
 - creates a message box displaying *message_string*
The box contains an 'OK' button that the user will have to click
(alternatively, the message box can be closed)
for the execution of the remaining code to proceed

Exam <https://eduassistpro.github.io>

```
alert(
```

Add WeChat edu_assist_pro

Window object: Dialog boxes

- `bool confirm(message_string)`
 - creates a message box displaying *message_string*
The box contains two buttons 'Cancel' and 'OK'
 - the function returns `true` if the user selects 'OK', `false` otherwise

Exam

```
var answer = confirm("Do you want to proceed?")
```

Add WeChat edu_assist_pro

Window object: Dialog boxes

- string `prompt(message_string, default)`

- creates a dialog box displaying

Example:

message_string and an
input field

var userName =

`prompt("What is your name?",`

- if a sec

is giv

sho

- the b

'Cancel' and 'OK'

- if the user selects 'OK' then

the current value entered in

the input field is returned as a

string, otherwise null is

returned

Add WeChat edu_assist_pro

Window object: Dialog boxes

- `prompt()` always returns a string, even if the user enters a number
- To convert a string to number the following functions can be used:

Assignment Project Exam Help

- `number parseInt(string [, base])`
 - converts *string* to an integer number wrt numeral system *base*
 - only
 - if the
- `number Number(string)`
 - converts *string* to a floating-point number
 - only converts up to the first invalid character in
 - if the first non-white-space character in
- `number Number(string)`
 - returns `Nan` if *string* contains an invalid character

Dialog boxes: Example

```
<html>
  <head><title>Interaction example</title></head>
<body>
<script type='text/javascript'>
do {
    string      = prompt("How many items do you want to buy?")
    quanti
} while (isNaN(string))
do {
    string = pr
    price   = parseFloat(string)
} while (isNaN(price) || price <= 0)
buy = confirm('You will have to pay '+
              (price*quantity).toFixed(2)+'
              '\nDo you want to proceed?')
if (buy) alert("Purchase made")
</script>
</body></html>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/jsPrompt.html>

User input validation

- A common use of JavaScript is the validation of user input in a HTML form before it is processed:

- check that required fields have not been left empty
- check that fields only contain allowed characters or com
- che

```
<form method="post" onsubmit="return validate(this)">
  <label>User name: <input type="text" name="user"></label>
  <label>Email address: <input type="text" name="email"></label>
  <input type="submit" name="submit" value="Submit" />
</form>
<script>
function validate(form) {
  var fail = validateUser(form.user.value)
  fail += validateEmail(form.email.value)
  if (fail == "") return true
  else { alert(fail); return false }
}</script>
```

User input validation

```
1 function validateUser(field) {  
2     if (field == "") return "No username entered\n"  
3     else if (field.length < 5)  
4         return "Username too short\n"  
5     else if (/[^a-zA-Z0-9_-]/.test(field))  
6         r  
7     else r  
8 }  
9  
10 function validateEmail(field) {  
11     if (field == "") return "No email entered\n"  
12     else if (!((field.indexOf(".") > 0) &&  
13                 (field.indexOf("@") > 0)) ||  
14                 /^[^a-zA-Z0-9._-]/.test(field))  
15         return "Invalid character in email\n"  
16     else return ""  
17 }
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/jsValidate.html>

Window and Document objects

JavaScript provides two objects that are essential to the creation of dynamic web pages and interactive web applications:

Assignment Project Exam Help

- an object-oriented representation of a web page (HTML document) that is displayed
- allows

<https://eduassistpro.github.io/>

Exam

Document Object Model

Add WeChat edu_assist_pro

A platform- and language-neutral interface that allows scripts to dynamically access and update the content, structure and style of HTML, XHTML and XML documents

Document Object Model

Example:

The HTML table below

```
<table>
  <tbody>
    <tr>
      <td>Shady Grove</td>
      <td>
        <br>
        Dorian
      </td>
    </tr>
    <tr>
      <td>
        <br>
        Dorian
      </td>
      <td>Dorian</td>
    </tr>
  </tbody>
</table>
```

is parsed into the following DOM

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

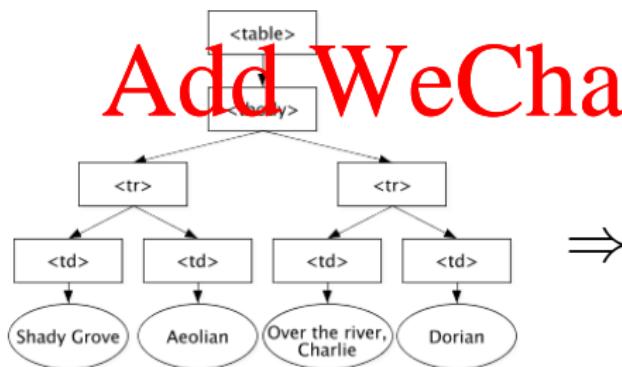
Arnaud Le Hors, et al, editors: Document Object Model (DOM) Level 3 Core Specification, Version 1.0, W3C Recommendation 07 April 2004. World Wide Web Consortium, 2004.
<https://www.w3.org/TR/DOM-Level-3-Core/> [accessed 9 January 2017]

Accessing HTML elements: Object methods

Example:

```
// access the tbody element from the table element  
var myTbodyElement = myTableElement.firstChild;  
// access its second tr element; the list of children starts at 0 (not 1).  
var mySecondTrElement = myTbodyElement.childNodes[1];  
  
// remove its first td el  
mySecondTrEl...  
// change the text con  
mySecondTrElement.firstChild.firstChild.data = "Peter";
```

Assignment Project Exam Help
<https://eduassistpro.github.io>



Accessing HTML elements: Names (1)

Instead of using methods such as `firstChild` and `childNodes[n]`, it is possible to assign `names` to denote the children of a HTML element

Example:

```
<form name="form1" action="">
<label>Temperature in Fahrenheit:</label>
<input type="text" name="celsius">
<label>Temperature in Celsius:</label>
<input type="text" name="fahrenheit">
</form>
```

<https://eduassistpro.github.io/>

Then – `document.form1`

– Refers to the whole form

– `document.form1.celsius`

– Refers to the text field named `celsius` in `document.form1`

– `document.form1.celsius.value`

– Refers to the attribute value in the text field named `celsius` in `document.form1`

Accessing HTML elements: Names (2)

Accessing HTML elements by giving them **names** and using **paths** within the Document Object Model tree structure is still problematic

If that tree structure changes, then those **paths** no longer work.

Assignment Project Exam Help

Example:

Change in

```
<form name="f">
<div class="f">
<label>Tem
<input type="text" name="fahrenheit" size=10 value="0" />
</div>
<div class="field" name="cdiv">
<label>Temperature in Celsius:</label>
<input type="text" name="celsius" size="10" value="" />
</div>
</form>
```

Add WeChat edu_assist_pro

means that `document.form1.celsius` no longer works as there is now a `div` element between `form` and `text field`, we would now need to use `document.form1.cdiv.celsius`

Accessing HTML elements: IDs

A more reliable way is to give each HTML element an ID
(using the `id` attribute) and to use `getElementById` to retrieve
an HTML element by its ID

Assignment Project Exam Help

Example:

```
<form id="for"
<label>Tem
<input type="text"
<label>Tem
<input type="text"
</form>
```

<https://eduassistpro.github.io/>

Then

– `document.getElementById('celsius')`

Refers to the HTML element with ID `celsius` in document

– `document.getElementById('celsius').value`

Refers to the attribute value in the HTML element with ID `celsius`
in document

Manipulating HTML elements

It is not only possible to access HTML elements, but also possible to change them on-the-fly

```
<html><head><title>Manipulating HTML elements</title>
<style>
  td.RedBG { background: #f00; }
</style>
<script>
function changeBackground1(id) {
  document.getElementById(id).style.backgroundColor = "red";
  document.getElementById(id).innerHTML = "red";
}
function changeBackground2(id) {
  document.getElementById(id).cell.className = "RedBG";
  document.getElementById(id).cell.innerHTML = "red";
}
</script></head><body>
<table border="1"><tr>
  <td id="0" onclick="changeBackground1('0');">white</td>
  <td id="1" onclick="changeBackground2('1');">white</td>
</tr></table></body></html>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/jsBG.html>

Event-driven JavaScript Programs

- The JavaScript programs we have seen so far were all **executed sequentially**

Assignment Project Exam Help

• programs have a particular starting point

- programs are executed step-by-step,
involving
- **problem** <https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Event-Driven JavaScript Programs

- Web applications are event-driven
 - ~ they react to events such as mouse clicks and key strokes

Assignment Project Exam Help

<https://eduassistpro.github.io/>

nickwalter's: What is Event Driven Programming?

SlideShare, 7 September 2014.

<https://tinyurl.com/ya58xbs9> [accessed 5/11/2017]

Add WeChat edu_assist_pro

- With JavaScript,
 - we can define event handler functions for a wide variety of events
 - event handler functions can manipulate the document object
(changing the web page in situ)

Event Handlers and HTML Elements

- HTML events are things, mostly user actions, that happen to HTML elements
- Event handlers are JavaScript functions that process events

Assignment Project Exam Help

- Event handlers must be associated with HTML elements for specific event
- This can be done by adding an attribute to the element:
`<input type="button" value="Click Me!" onclick="Hello()>`
- Alternatively, a JavaScript function can be used to associate an event handler with an HTML element

Add WeChat edu_assist_pro

```
// All good browsers
window.addEventListener("load", Hello)
// MS IE browser
window.attachEvent("onload", Hello)
```

More than one event handler can be added this way to the same element for the same event

Event Handlers and HTML Elements

- As our scripts should work with as many browsers as possible, we need to detect which method works:

```
if (window.addEventListener) {
    window.addEventListener("load", Hello)
} else {
    wi
}
```

- Event h

```
if (window.removeEventListener) {
    window.removeEventListener("load", Hello)
} else {
    window.detachEvent("onload", Hello)
}
```

Add WeChat edu_assist_p

Events: Load

- An `(on)load` event occurs when an object has been loaded
- Typically, event handlers for `onload` events are associated with the `window` object or the `body` element of an HTML document

```
<html>
  <head>
    <ti
    <sc
      </script>
    </head>
    <body onload="Hello ()">
      <p>Content of the web page</p>
    </body>
</html>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP519/examples/jsOnload.html>

Events: Focus / Change

- A **focus event** occurs when a form field receives input focus by tabbing with the keyboard or clicking with the mouse

→ `onFocus` attribute

- A **change event** occurs when a select, text, or textarea field loses focus and its value changes

→ `on`

Example <https://eduassistpro.github.io/>

```
<form name="form1" method="post" action="process.php">
  <select name="select" required
    onChange="document.form1.submit();"
    <option value="">Select a name</option>
    <option value="200812345">Tom Beck</option>
    <option value="200867890">Jim Kent</option>
  </select>
</form>
```

Add WeChat edu_assist_program

Events: Focus / Change

- A **focus event** occurs when a form field receives input focus by tabbing with the keyboard or clicking with the mouse

onFocus attribute

- A **change event** occurs when a select, text, or textarea field loses focus and its value has been modified

~ on

<https://eduassistpro.github.io/Assignment%20Project%20Exam%20Help.html>

```
<form>
  <label>Temperature in Fahrenheit:</label>
  <input type="text" id="fahrenheit" size="10" value="0"
    onchange="document.getElementById('celsius').value =
      FahrenheitToCelsius(parseFloat(document.getElementById('fahrenheit').value))">
  <br>
  <label>Temperature in Celsius:</label>
  <input type="text" id="celsius"
    size="10" value="" onfocus="blur();"/>
</form>
```

<http://cgi.csc.liv.ac.uk/~ullrich/COMP519/examples/jsOnchange.html>

Events: Blur / Click

- A **blur event** occurs when an HTML element loses focus
 - ~ `onBlur` attribute
- A **click event** occurs when an object on a form is clicked
 - ~ `onClick` attribute

Example

```
<html><he  
<form name=""  
    Enter a num  
    <input type="text" size="12" id="number" value="3.1">  
    <br><br>  
    <input type="button" value="Double"  
        onclick="document.getElementById('number').v  
            parseFloat(document.getElementById('number').value)  
            * 2;">  
</form></body></html>
```

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

<http://cgi.csc.liv.ac.uk/~ullrich/COMP284/examples/jsOnlick.html>

Events: MouseOver / Select / Submit

- A **keydown event** occurs when the user presses a key

 ~**onkeydown** attribute

- A **MOUSEOVER event** occurs once each time the mouse pointer moves over an HTML element from outside that element

 ~**on**

- A **select** or text

 ~**onSelect** attribute

- A **submit event** occurs when a user submits a form

 ~**onSubmit** attribute

Assignment Project Exam Help
<https://eduassistpro.github.io>
Add WeChat edu_assist_pro

Events and DOM

- When an event occurs, an event object is created
 - ~ an event object has attributes and methods
 - ~ event objects can be created by your code independent of an event occurring

Assignment Project Exam Help

- In most as an arg
- In most can only

<https://eduassistpro.github.io>

```
<html><body onKeyDown="processKey(event)">
  <script>
    function processKey(e) {
      e = e || window.event
      document.getElementById("key").innerHTML =
        String.fromCharCode(e.keyCode) + ' has been pressed'
    }
  </script>
  <!-- key code will appear in the paragraph below -->
  <p id="key"></p>
</body></html>
```

Revision

Read

- Chapter 17: JavaScript and PHP Validation and Error Handling
- Chapter 18: Using Ajax

Assignment Project Exam Help

of
R. Nixon:

Learnin
O'Reilly

<https://eduassistpro.github.io>

- Mozilla Developer Network and individual components of the Document Object Model (DOM). 18 March 2014. <https://developer.mozilla.org> [accessed 18 March 2014].
- W3Schools: JavaScript and HTML DOM Reference, 18 March 2014. <http://www.w3schools.com/jsref/> [accessed 18 March 2014].

Add WeChat edu_assist_pro