

!= (Logical operator not)	
<i>lecture01</i>	p. 23
" (Double quote)	
<i>lecture01</i>	p. 15
#define	
<i>lecture01</i>	p. 16,18
<i>lecture09</i>	p. 3,4
#ifndef	
<i>lecture09</i>	p. 3
#include	
<i>lecture01</i>	p. 16,20
<i>lecture02</i>	p. 7
<i>lecture09</i>	p. 3
%p	
<i>lecture09</i>	p. 22
& (bit and)	
<i>lecture01</i>	p. 23
& operator (address)	
<i>lecture01</i>	p. 10
<i>lecture03</i>	p. 4,6,8
&& (Logical operator and)	
<i>lecture01</i>	p. 23
' (Single quote)	
<i>lecture01</i>	p. 15
* operator (dereferencing)	
<i>lecture01</i>	p. 10
<i>lecture03</i>	p. 7
- (arrow) reference to structure	
<i>lecture03</i>	p. 19,20
. (dot) reference to structure	
<i>lecture03</i>	p. 16,20
.h File	
<i>lecture01</i>	p. 16
.hpp file	
<i>lecture08</i>	p. 14
2-3-4 Tree	
<i>lecture06</i>	p. 20
\0	
<i>lecture01</i>	p. 14,15
<i>lecture02</i>	p. 10-12
\n	
<i>lecture02</i>	p. 10

<b>__FILE__</b>	
<i>lecture09</i>	p. 5,6
<b>__LINE__</b>	
<i>lecture09</i>	p. 5,6
((Curly brackets)	
<i>lecture01</i>	p. 22
(bit or)	
<i>lecture01</i>	p. 23
(Logical operator or)	
<i>lecture01</i>	p. 23

## A

Address	
<i>lecture01</i>	p. 8-10,12,13
Address of variable	
<i>lecture03</i>	p. 4
Adelson-Velsky, Georgy	
<i>lecture06</i>	p. 17
Algorithm	
<i>lecture01</i>	p. 12
Alignment of structures	
<i>lecture03</i>	p. 18
And (logical operator)	
<i>lecture01</i>	p. 23
Angle brackets vs double quotes for header files	
<i>lecture02</i>	p. 7
Architecture	
<i>lecture01</i>	p. 10
<b>argc</b>	
<i>lecture02</i>	p. 2
<i>lecture03</i>	p. 14,15
<i>lecture03</i>	p. 13
Argument passed as reference	
<i>lecture08</i>	p. 16
Argument passed by reference	
<i>lecture08</i>	p. 17
<b>argv[ ]</b>	
<i>lecture02</i>	p. 2
<i>lecture03</i>	p. 13-15
Array	
<i>lecture01</i>	p. 10,12-14
<i>lecture03</i>	p. 6

<i>lecture05</i>	p. 17-21
<i>lecture06</i>	p. 1,9,10
<i>lecture08</i>	p. 20
Array in C and in Java	
<i>lecture03</i>	p. 12
Array of strings	
<i>lecture03</i>	p. 12,13
Array of structures	
<i>lecture03</i>	p. 16
Array vs pointer	
<i>lecture03</i>	p. 6,8,9,12
Array – multidimensional	
<i>lecture03</i>	p. 13
Array: returned by a function	
<i>lecture04</i>	p. 11-13
Array:Pointer	
<i>lecture05</i>	p. 2
Arrow reference to structure field	
<i>lecture03</i>	p. 19,20
ASCII	
<i>lecture01</i>	p. 9
ASCII table	
<i>lecture01</i>	p. 22
<b>assert</b>	
<i>lecture02</i>	p. 4
Assigning address to pointer	
<i>lecture03</i>	p. 6
Assignment	
<i>lecture01</i>	p. 22,23
<i>lecture02</i>	p. 2
ATK	
<i>lecture09</i>	p. 18
<b>atof()</b>	
<i>lecture03</i>	p. 1
<b>atoi()</b>	
<i>lecture03</i>	p. 1
<b>atol()</b>	
<i>lecture03</i>	p. 1
<b>autoconf</b>	
<i>lecture09</i>	p. 11
<b>automake</b>	
<i>lecture09</i>	p. 11
<b>autoscan</b>	
<i>lecture09</i>	p. 10
Autotools	

<i>lecture09</i>	p. 9-11
AVL tree	
<i>lecture06</i>	p. 17-19

## B

B-Tree	
<i>lecture06</i>	p. 20-23
<i>lecture07</i>	p. 2
Balanced tree	
<i>lecture06</i>	p. 17-19
Bell Labs	
<i>lecture01</i>	p. 6,7
Berkeley Software Distribution (BSD)	
<i>lecture07</i>	p. 2
Binary file	
<i>lecture04</i>	p. 2
Binary search	
<i>lecture06</i>	p. 9,10,13,14
Binary tree	
<i>lecture06</i>	p. 14-17
Bit	
<i>lecture01</i>	p. 8
Bit operators	
<i>lecture01</i>	p. 23
Block	
<i>lecture01</i>	p. 22
Block of instructions	
<i>lecture01</i>	p. 15
Boolean	
<i>lecture01</i>	p. 10,11,21
Box-Müller	
<i>lecture03</i>	p. 3
<b>break</b>	
<i>lecture01</i>	p. 24
BSD (Berkeley Software Distribution)	
<i>lecture07</i>	p. 2
Built-in functions	
<i>lecture02</i>	p. 5,8
Byte	
<i>lecture01</i>	p. 8

## C

C environment	
<i>lecture01</i>	p. 7
C program structure	
<i>lecture01</i>	p. 18
C standard library	
<i>lecture02</i>	p. 8
C vs Java	
<i>lecture01</i>	p. 6,12-14,17,18
<i>lecture02</i>	p. 5
<i>lecture03</i>	p. 12,20
<i>lecture04</i>	p. 7,17
<i>lecture05</i>	p. 1,2,16
<i>lecture06</i>	p. 23
<i>lecture09</i>	p. 15-17
C++	
<i>lecture08</i>	p. 10-13
C++ vs Java	
<i>lecture08</i>	p. 18-20
C11	
<i>lecture01</i>	p. 7
C89	
<i>lecture01</i>	p. 7
C99	
<i>lecture01</i>	p. 7
Cairo	
<i>lecture09</i>	p. 18
Calling functions	
<i>lecture04</i>	p. 8
<b>calloc()</b>	
<i>lecture04</i>	p. 18
<b>case</b>	
<i>lecture01</i>	p. 24
Case	
<i>lecture02</i>	p. 11
Case insensitive comparison	
<i>lecture02</i>	p. 13
<b>catch</b>	
<i>lecture08</i>	p. 16
Catching errors	
<i>lecture08</i>	p. 11,12
CFLAGS	
<i>lecture04</i>	p. 5
Changing case	
<i>lecture02</i>	p. 11

<b>char</b>	
<i>lecture01</i>	p. 10,11
Character classification	
<i>lecture02</i>	p. 10,11
Character encoding	
<i>lecture01</i>	p. 15,22
Character Encoding	
<i>lecture02</i>	p. 17
Character encoding	
<i>lecture02</i>	p. 15-18
Chinese characters	
<i>lecture02</i>	p. 14,16,17
<b>cin</b>	
<i>lecture08</i>	p. 15
CJK	
<i>lecture02</i>	p. 15
<b>class</b>	
<i>lecture08</i>	p. 18
<i>lecture09</i>	p. 15
Classes	
<i>lecture08</i>	p. 17,18
Classification of characters	
<i>lecture02</i>	p. 10,11
Code	
<i>lecture01</i>	p. 8
<b>codepoint</b>	
<i>lecture02</i>	p. 15
<i>lecture02</i>	p. 15
Collection	
<i>lecture08</i>	p. 19
Collections	
<i>lecture05</i>	p. 16
Command-line parameters	
<i>lecture02</i>	p. 2
<i>lecture03</i>	p. 13-15
Comparison of Data structures	
<i>lecture07</i>	p. 3-5
Comparison of strings	
<i>lecture02</i>	p. 12,13
Comparison operators	
<i>lecture01</i>	p. 22,23
<i>lecture02</i>	p. 2
Compiler	
<i>lecture01</i>	p. 16,17,19,20
Compiling a C program	

<i>lecture01</i>	p. 17
Compiling on Linux	
<i>lecture01</i>	p. 17
Condition	
<i>lecture01</i>	p. 21
Conditional compiling	
<i>lecture09</i>	p. 6,9
<b>configure</b>	
<i>lecture09</i>	p. 10,11
<b>conio.h</b>	
<i>lecture09</i>	p. 8
Constants	
<i>lecture01</i>	p. 16,18
Constructor	
<i>lecture08</i>	p. 18
Course expectations	
<i>lecture01</i>	p. 2
Course notes	
<i>lecture01</i>	p. 3
Course Organization	
<i>lecture01</i>	p. 6
Course schedule	
<i>lecture01</i>	p. 1
<b>cout</b>	
<i>lecture08</i>	p. 14,15
Craftsmanship	
<i>lecture01</i>	p. 5
Cryptography	
<i>lecture06</i>	p. 12
<b>ctime()</b>	
<i>lecture03</i>	p. 2,3,21
<b>ctype.h</b>	
<i>lecture02</i>	p. 10
<i>lecture02</i>	p. 11
Curly brackets	
<i>lecture01</i>	p. 15,22
Cygwin	
<i>lecture01</i>	p. 2

## D

Dahl, Ole-Johan	
<i>lecture08</i>	p. 12
Data	

<i>lecture01</i>	p. 8
<i>lecture05</i>	p. 16,17
Data structure	
<i>lecture05</i>	p. 17
Data structure functions	
<i>lecture06</i>	p. 23
<i>lecture07</i>	p. 2,3
Data structures	
<i>lecture05</i>	p. 16,17,21,22
<i>lecture06</i>	p. 1-13
<i>lecture07</i>	p. 4,5
Data structures comparison	
<i>lecture07</i>	p. 3-5
Data types	
<i>lecture01</i>	p. 10-12
Database	
<i>lecture06</i>	p. 20,23
<i>lecture07</i>	p. 5
<b>ddd</b>	
<i>lecture09</i>	p. 22
Debugging	
<i>lecture09</i>	p. 22
Declaration	
<i>lecture05</i>	p. 1
Declaration of pointer	
<i>lecture03</i>	p. 5,6
Declaration of variable	
<i>lecture01</i>	p. 9
Default parameters	
<i>lecture08</i>	p. 16
Degenarated binary tree	
<i>lecture06</i>	p. 17
<b>delete</b>	
<i>lecture08</i>	p. 15
Deleting a file	
<i>lecture04</i>	p. 2
Dereferencing	
<i>lecture03</i>	p. 7,8,19,20
Destructor	
<i>lecture08</i>	p. 18
Direct access	
<i>lecture04</i>	p. 2
Directory operations	
<i>lecture04</i>	p. 3
<b>dirent.h</b>	

<i>lecture04</i>	p. 3
Distribution	
<i>lecture03</i>	p. 3
<b>do ... while</b>	
<i>lecture02</i>	p. 1
Dot reference to structure field	
<i>lecture03</i>	p. 16,20
<b>double</b>	
<i>lecture01</i>	p. 12
Double quote	
<i>lecture01</i>	p. 15
Double quotes vs angle brackets for header files	
<i>lecture02</i>	p. 7
Doubly linked list	
<i>lecture06</i>	p. 8
Dumping a binary file	
<i>lecture04</i>	p. 3
Dynamic analysis:gdb	
<i>lecture09</i>	p. 22
Dynamic analysis:Valgrind	
<i>lecture09</i>	p. 22
Dynamic data structures	
<i>lecture07</i>	p. 5
Dynamic memory	
<i>lecture04</i>	p. 16-18
<i>lecture05</i>	p. 1,2,17-20
Dynamic memory example	
<i>lecture04</i>	p. 19

## E

Eclipse	
<i>lecture09</i>	p. 22
EDP	
<i>lecture05</i>	p. 16
Electric-Fence	
<i>lecture09</i>	p. 22
Electronic Data Processing	
<i>lecture05</i>	p. 16
<b>else</b>	
<i>lecture01</i>	p. 21,23
<b>else if</b>	
<i>lecture01</i>	p. 23

Encapsulation	
<i>lecture08</i>	p. 18
Encoding	
<i>lecture01</i>	p. 9
<i>lecture02</i>	p. 15
End-of-string marker	
<i>lecture01</i>	p. 14,15
<b>EOF</b>	
<i>lecture02</i>	p. 9,10
Epoch	
<i>lecture03</i>	p. 2
<b>errno</b>	
<i>lecture03</i>	p. 1
<i>lecture09</i>	p. 15
<b>errno.h</b>	
<i>lecture03</i>	p. 1
<i>lecture09</i>	p. 15
Error checking	
<i>lecture02</i>	p. 2-4
<i>lecture03</i>	p. 1
Error management	
<i>lecture02</i>	p. 4,5
Exam	
<i>lecture01</i>	p. 3
Exam dates	
<i>lecture01</i>	p. 3
Example of pointer usage	
<i>lecture03</i>	p. 8
Example: day of the week when you were born	
<i>lecture03</i>	p. 21-23
Example: linked list	
<i>lecture06</i>	p. 4-6
Exams	
<i>lecture01</i>	p. 2-4
Exception	
<i>lecture02</i>	p. 4,5
Exceptions	
<i>lecture08</i>	p. 16
Executable	
<i>lecture01</i>	p. 16
Expectations	
<i>lecture01</i>	p. 2
Exponent	
<i>lecture01</i>	p. 12
Exponential distribution	

<i>lecture03</i>	p. 3
<b>extern</b>	
<i>lecture04</i>	p. 7
<i>lecture04</i>	p. 7
<i>lecture09</i>	p. 14

## F

Factorial	
<i>lecture05</i>	p. 14
<b>fclose()</b>	
<i>lecture03</i>	p. 25
<b>fepf()</b>	
<i>lecture04</i>	p. 1
<b>ferror()</b>	
<i>lecture04</i>	p. 1
<b>fflush()</b>	
<i>lecture09</i>	p. 22
<b>fgetc()</b>	
<i>lecture02</i>	p. 9
<i>lecture04</i>	p. 1
<b>fgets()</b>	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 10
<i>lecture04</i>	p. 14
fgets():Return value	
<i>lecture03</i>	p. 1
FIFO	
<i>lecture06</i>	p. 8
<i>lecture07</i>	p. 4
<b>FILE</b>	
<i>lecture03</i>	p. 26
<b>FILE *</b>	
<i>lecture03</i>	p. 25
Files	
<i>lecture03</i>	p. 24-26
Final exam	
<i>lecture01</i>	p. 3
First In First Out	
<i>lecture06</i>	p. 8
<b>float</b>	
<i>lecture01</i>	p. 12
<b>flock()</b>	
<i>lecture04</i>	p. 2

Flow control	
<i>lecture01</i>	p. 21,23,24
<i>lecture02</i>	p. 1
<b>fopen()</b>	
<i>lecture03</i>	p. 25,26
<b>for</b>	
<i>lecture02</i>	p. 1
Formatted input and output	
<i>lecture02</i>	p. 10
<b>fprint()</b>	
<i>lecture03</i>	p. 26
<b>fprintf()</b>	
<i>lecture02</i>	p. 10
<b>fputc()</b>	
<i>lecture02</i>	p. 9
<i>lecture04</i>	p. 1
<b>fputs()</b>	
<i>lecture02</i>	p. 10
<i>lecture03</i>	p. 26
<b>fread()</b>	
<i>lecture04</i>	p. 1
Free Software Foundation (FSF)	
<i>lecture09</i>	p. 9
<b>free()</b>	
<i>lecture04</i>	p. 19
<i>lecture04</i>	p. 18,21-23
Freeing a binary tree	
<i>lecture06</i>	p. 16
<b>fseek()</b>	
<i>lecture04</i>	p. 2
FSF	
<i>lecture09</i>	p. 9
Function call	
<i>lecture04</i>	p. 8-12
Function declaration	
<i>lecture02</i>	p. 6,7
Function identification	
<i>lecture02</i>	p. 5,6
Function nesting	
<i>lecture01</i>	p. 15
Function pointer	
<i>lecture06</i>	p. 23,24
Function pointers	
<i>lecture09</i>	p. 16,17
Function prototype	

<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 7
Function: Pointers as argument	
<i>lecture04</i>	p. 13,14
<i>lecture05</i>	p. 2,3
Function: returning an array	
<i>lecture04</i>	p. 11-13
Functions	
<i>lecture04</i>	p. 8
Functions, nesting	
<i>lecture02</i>	p. 6
<b>fwrite()</b>	
<i>lecture04</i>	p. 1

## G

<b>g++</b>	
<i>lecture08</i>	p. 15
Garbage collector	
<i>lecture04</i>	p. 19
<i>lecture08</i>	p. 19
<b>gcc</b>	
<i>lecture01</i>	p. 17
<i>lecture01</i>	p. 7
<b>gcd()</b>	
<i>lecture04</i>	p. 8,9
<b>getchar()</b>	
<i>lecture02</i>	p. 9
<b>getopt()</b>	
<i>lecture03</i>	p. 15
<b>gets()</b>	
<i>lecture02</i>	p. 10
Git	
<i>lecture09</i>	p. 21
Glib	
<i>lecture07</i>	p. 3
<i>lecture09</i>	p. 18
Global variable	
<i>lecture03</i>	p. 1
<i>lecture04</i>	p. 15
<i>lecture09</i>	p. 13,15,16
<b>gmtime()</b>	
<i>lecture03</i>	p. 21
Gnome	

<i>lecture07</i>	p. 3
Gnome Tool Kit (GTK)	
<i>lecture09</i>	p. 18-20
GNU	
<i>lecture07</i>	p. 3
GNU autotools	
<i>lecture09</i>	p. 9-11
Grades	
<i>lecture01</i>	p. 4
GTK (Gnome Tool Kit)	
<i>lecture09</i>	p. 18-20
<b>gtk.h</b>	
<i>lecture09</i>	p. 18
<b>GtkWidget</b>	
<i>lecture09</i>	p. 18,19
<b>GTK_WINDOW</b>	
<i>lecture09</i>	p. 19

## H

Hanoi (towers of)	
<i>lecture05</i>	p. 15
Hash function	
<i>lecture06</i>	p. 11,12
Hash table	
<i>lecture06</i>	p. 11-13
<i>lecture07</i>	p. 2
<b>head</b>	
<i>lecture04</i>	p. 3
Head of list	
<i>lecture06</i>	p. 1
Header file	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 7
<i>lecture09</i>	p. 1,2,15
Heap	
<i>lecture01</i>	p. 8
<i>lecture04</i>	p. 17
Help on functions	
<i>lecture02</i>	p. 5
History of C	
<i>lecture01</i>	p. 7
Hoare, Antony	
<i>lecture05</i>	p. 6

Honesty  
*lecture01* p. 5

## I

**if**  
*lecture01* p. 21,23  
In-memory database  
*lecture07* p. 5  
Information  
*lecture05* p. 16,17  
Information Technology  
*lecture05* p. 16  
Initialization of pointer  
*lecture03* p. 7,8  
Initialization of structure  
*lecture03* p. 16  
Input/Output  
*lecture02* p. 9,10  
Insertion in a binary tree  
*lecture06* p. 15,16

**int**  
*lecture01* p. 11

**integer operations**  
*lecture01* p. 11

**iostream**  
*lecture08* p. 14

**isalnum()**  
*lecture02* p. 11

**isalpha()**  
*lecture02* p. 11

**isdigit()**  
*lecture02* p. 11

**islower()**  
*lecture02* p. 11

ISO  
*lecture02* p. 16

**isprint()**  
*lecture02* p. 11

**ispunct()**  
*lecture02* p. 11

**isspace()**  
*lecture02* p. 11

**isupper()**

*lecture02* p. 11  
IT  
*lecture05* p. 16

## J

Java vs C  
*lecture01* p. 6,12-14,17,18  
*lecture02* p. 5  
*lecture03* p. 12,20  
*lecture04* p. 7  
java vs C  
*lecture04* p. 17  
Java vs C  
*lecture05* p. 1,2,16  
*lecture06* p. 23  
*lecture09* p. 15-17  
Java vs C++  
*lecture08* p. 18-20

## K

K&R  
*lecture01* p. 6  
Keringhan (Brian)  
*lecture01* p. 6

## L

Lab2 hints  
*lecture05* p. 3-5  
Labs  
*lecture01* p. 3,4  
Landis, Evgenii  
*lecture06* p. 17  
Last In First Out  
*lecture06* p. 7,8  
**ld**  
*lecture01* p. 20  
Library file  
*lecture04* p. 6  
LIFO



<i>lecture06</i>	p. 7,8
<i>lecture07</i>	p. 4
Linked list	
<i>lecture05</i>	p. 22
<i>lecture06</i>	p. 1-10,13
<i>lecture07</i>	p. 1
Linker	
<i>lecture01</i>	p. 16,17,19,20
<i>lecture04</i>	p. 7-9
Linux	
<i>lecture01</i>	p. 2
List	
<i>lecture08</i>	p. 19
<b>localtime()</b>	
<i>lecture03</i>	p. 21
Locking a file	
<i>lecture04</i>	p. 2
Logical operators	
<i>lecture01</i>	p. 23
<b>long</b>	
<i>lecture01</i>	p. 11
<i>lecture01</i>	p. 11
Loop	
<i>lecture02</i>	p. 1

## M

Macro	
<i>lecture09</i>	p. 4,5
<b>main()</b>	
<i>lecture01</i>	p. 16
<b>make</b>	
<i>lecture01</i>	p. 7
<i>lecture04</i>	p. 5,6
<i>lecture04</i>	p. 4-6
<i>lecture09</i>	p. 9
Makefile	
<i>lecture04</i>	p. 5,6
<b>malloc()</b>	
<i>lecture04</i>	p. 18-20
<i>lecture05</i>	p. 20
<i>lecture09</i>	p. 16
<b>man</b>	
<i>lecture02</i>	p. 5

Marker (end-of-string)	
<i>lecture01</i>	p. 14,15
Mathematical functions	
<i>lecture01</i>	p. 19,20
Mathematical functions:Compiler	
<i>lecture01</i>	p. 19
Mathematical Induction	
<i>lecture05</i>	p. 10
Mathematical induction	
<i>lecture05</i>	p. 8-10
Matrix example	
<i>lecture09</i>	p. 2,3
Maurolico, Francisco	
<i>lecture05</i>	p. 9
<b>MD5</b>	
<i>lecture06</i>	p. 12
<b>memory</b>	
<i>lecture01</i>	p. 8
<i>lecture01</i>	p. 8
Memory address	
<i>lecture01</i>	p. 9,10
Memory leak	
<i>lecture04</i>	p. 22
Mercurial	
<i>lecture09</i>	p. 21
Method	
<i>lecture06</i>	p. 24
Method definition	
<i>lecture08</i>	p. 18
Methods	
<i>lecture08</i>	p. 17
<i>lecture09</i>	p. 1
Methods in structures	
<i>lecture08</i>	p. 18
Midcourse exam	
<i>lecture01</i>	p. 3
MidCourse Exam	
<i>lecture08</i>	p. 1-10
<b>mktime()</b>	
<i>lecture03</i>	p. 21
Multi-threading	
<i>lecture04</i>	p. 15,16
Multidimensional array	
<i>lecture03</i>	p. 13
Multiple inclusions	

<i>lecture09</i>	p. 3
<b>N</b>	
Name of variable	
<i>lecture01</i>	p. 9
<b>namespace</b>	
<i>lecture08</i>	p. 14
Naming a structure	
<i>lecture03</i>	p. 16,17
Nesting functions	
<i>lecture02</i>	p. 6
<b>new</b>	
<i>lecture08</i>	p. 15
<i>lecture09</i>	p. 16
<b>nm</b>	
<i>lecture09</i>	p. 14
Node	
<i>lecture05</i>	p. 21,22
Non binary tree	
<i>lecture06</i>	p. 20-23
Normal distribution	
<i>lecture03</i>	p. 3
Not (logical operator)	
<i>lecture01</i>	p. 23
<b>NULL</b>	
<i>lecture02</i>	p. 10,13
<i>lecture03</i>	p. 1,7
Nygaard, Kristen	
<i>lecture08</i>	p. 12
Nygard, Kirsten	
<i>lecture08</i>	p. 12

## O

Object	
<i>lecture08</i>	p. 19
Object Oriented Programming	
<i>lecture09</i>	p. 17
Object reference	
<i>lecture08</i>	p. 19
Object-Oriented Programming	
<i>lecture06</i>	p. 24

<b>od</b>	
<i>lecture04</i>	p. 3
<b>operator</b>	
<i>lecture08</i>	p. 14,15
Or (logical operator)	
<i>lecture01</i>	p. 23
Order	
<i>lecture05</i>	p. 20,21
<i>lecture07</i>	p. 5
Over-engineering	
<i>lecture07</i>	p. 5
Overflow	
<i>lecture02</i>	p. 12
Overloading	
<i>lecture02</i>	p. 5
<i>lecture08</i>	p. 16

## P

Pango	
<i>lecture09</i>	p. 18
Pascal, Blaise	
<i>lecture05</i>	p. 9
<b>perror()</b>	
<i>lecture03</i>	p. 1
Persistence	
<i>lecture07</i>	p. 6
Pipe	
<i>lecture02</i>	p. 9
Pivot	
<i>lecture05</i>	p. 6-8
Pointer	
<i>lecture01</i>	p. 10
<i>lecture03</i>	p. 4-8,19,20
<i>lecture05</i>	p. 1,2
Pointer arithmetic	
<i>lecture03</i>	p. 10,11
Pointer on a function	
<i>lecture06</i>	p. 23,24
Pointer on structure	
<i>lecture03</i>	p. 19,20
Pointer to a file	
<i>lecture03</i>	p. 25
Pointer vs array	

<i>lecture03</i>	p. 6,8,9,12
Pointers	
<i>lecture04</i>	p. 11,12
Pointers as arguments to a function	
<i>lecture04</i>	p. 13,14
<i>lecture05</i>	p. 2,3
Pointers as parameters	
<i>lecture06</i>	p. 2,3
Pointers to functions	
<i>lecture09</i>	p. 16,17
Portability	
<i>lecture09</i>	p. 6-9
<b>pptx</b>	
<i>lecture04</i>	p. 3
Preprocessor	
<i>lecture01</i>	p. 16-18,20
<i>lecture09</i>	p. 3-6,8,9
<b>printf()</b>	
<i>lecture02</i>	p. 8,10
Priorities	
<i>lecture06</i>	p. 8
Project	
<i>lecture09</i>	p. 1
Prototype (function)	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 7
<b>public</b>	
<i>lecture08</i>	p. 18
<b>putchar()</b>	
<i>lecture02</i>	p. 9
<b>puts()</b>	
<i>lecture02</i>	p. 10

## Q

Quality	
<i>lecture01</i>	p. 5
Quick-sort	
<i>lecture05</i>	p. 6-8,11-14
Quiz 1	
<i>lecture07</i>	p. 6-8

## R

Radix	
<i>lecture01</i>	p. 12
<b>random()</b>	
<i>lecture03</i>	p. 2,3
Reading ZIP or XML	
<i>lecture04</i>	p. 3
<b>realloc()</b>	
<i>lecture04</i>	p. 18
<i>lecture05</i>	p. 19,20
Recursion	
<i>lecture05</i>	p. 10-15
<i>lecture06</i>	p. 5,6
Recursion vs loops	
<i>lecture05</i>	p. 14
Reference	
<i>lecture03</i>	p. 7
Reference to structure filed	
<i>lecture03</i>	p. 16
Return value	
<i>lecture02</i>	p. 3,4,8
Return value from main()	
<i>lecture01</i>	p. 16
Ritchie (Dennis)	
<i>lecture01</i>	p. 6,7
Ritchie, Dennis	
<i>lecture01</i>	p. 6
Robustness	
<i>lecture01</i>	p. 5
Root	
<i>lecture06</i>	p. 14
Rounding error	
<i>lecture01</i>	p. 12

## S

<b>scanf()</b>	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 3,4,10
<i>lecture04</i>	p. 14
SCCS	
<i>lecture09</i>	p. 21
Schedule	
<i>lecture01</i>	p. 1

Search		<i>lecture04</i>	p. 7,16
<i>lecture06</i>	p. 9,10	<i>lecture09</i>	p. 14
<i>lecture07</i>	p. 5	<i>lecture09</i>	p. 13,14
<b>search.h</b>		Static analysis:oclint	
<i>lecture07</i>	p. 2	<i>lecture09</i>	p. 22
Self-managing list		Static function	
<i>lecture06</i>	p. 8	<i>lecture09</i>	p. 14
Semi-colon		Static variable	
<i>lecture01</i>	p. 15	<i>lecture04</i>	p. 16
<b>setlocale</b>		<b>std</b>	
<i>lecture02</i>	p. 15	<i>lecture08</i>	p. 14
<b>setlocale()</b>		<b>stderr</b>	
<i>lecture03</i>	p. 3	<i>lecture02</i>	p. 9
<b>SHA1</b>		<i>lecture02</i>	p. 9
<i>lecture06</i>	p. 12	<i>lecture03</i>	p. 1
Shared library		<b>stdin</b>	
<i>lecture04</i>	p. 7	<i>lecture02</i>	p. 9
<b>short</b>		<i>lecture02</i>	p. 9,10
<i>lecture01</i>	p. 11	<i>lecture03</i>	p. 24,25
Side-effects		<b>stdio.h</b>	
<i>lecture09</i>	p. 5	<i>lecture03</i>	p. 25
<b>signed</b>		<b>stdlib.h</b>	
<i>lecture01</i>	p. 11,12	<i>lecture03</i>	p. 1
Simula		<i>lecture04</i>	p. 18
<i>lecture08</i>	p. 12	<b>stdout</b>	
Single quote		<i>lecture02</i>	p. 9,10
<i>lecture01</i>	p. 15	<i>lecture02</i>	p. 9,10
<b>sizeof()</b>		<i>lecture03</i>	p. 24,25
<i>lecture03</i>	p. 6,13	Strategy	
Sorting		<i>lecture06</i>	p. 7,8
<i>lecture05</i>	p. 6-8,11-14,19	<b>strcasecmp()</b>	
Source control		<i>lecture02</i>	p. 13
<i>lecture09</i>	p. 20,21	<b>strcat()</b>	
Splitting code		<i>lecture02</i>	p. 12
<i>lecture09</i>	p. 11,12	<b>strchr()</b>	
<b>sscanf()</b>		<i>lecture02</i>	p. 13
<i>lecture01</i>	p. 16	<b>strcmp()</b>	
Stack		<i>lecture02</i>	p. 12,13
<i>lecture01</i>	p. 8	<b>strcpy()</b>	
<i>lecture04</i>	p. 9-12	<i>lecture02</i>	p. 12
Stallman, Richard		<b>strdup()</b>	
<i>lecture09</i>	p. 9	<i>lecture04</i>	p. 18
Standard C++ library		<i>lecture05</i>	p. 18
<i>lecture08</i>	p. 15	Stream	
<b>static</b>		<i>lecture02</i>	p. 9

Stream redirection		<i>lecture02</i>	p. 13,14
<i>lecture03</i>	p. 24	<b>strtol()</b>	
<b>strerror()</b>		<i>lecture03</i>	p. 1
<i>lecture03</i>	p. 1	Struct	
String		<i>lecture03</i>	p. 20
<i>lecture01</i>	p. 10,14,15	<b>struct</b>	
<b>string</b>		<i>lecture03</i>	p. 15,17,20,23
<i>lecture08</i>	p. 15	<i>lecture03</i>	p. 16-20
String array		<i>lecture05</i>	p. 17
<i>lecture03</i>	p. 12,13	<i>lecture09</i>	p. 2
String comparison		struct (C++)	
<i>lecture02</i>	p. 12,13	<i>lecture08</i>	p. 17
String conversion to number		<b>struct tm</b>	
<i>lecture03</i>	p. 1	<i>lecture03</i>	p. 21
String declaration		Structure alignment	
<i>lecture03</i>	p. 11	<i>lecture03</i>	p. 18
String search		Structure and pointer	
<i>lecture02</i>	p. 13	<i>lecture03</i>	p. 19,20
<b>string.h</b>		Structure initialization	
<i>lecture02</i>	p. 11-13	<i>lecture03</i>	p. 16
<i>lecture04</i>	p. 18	Structure naming	
Strings		<i>lecture03</i>	p. 16,17
<i>lecture02</i>	p. 11-13	Structures	
<i>lecture03</i>	p. 1	<i>lecture03</i>	p. 15-20
<b>strlen()</b>		Subversion	
<i>lecture02</i>	p. 11	<i>lecture09</i>	p. 21
<b>strncasecmp()</b>		<b>switch</b>	
<i>lecture02</i>	p. 13	<i>lecture01</i>	p. 24
<b>strncat()</b>			
<i>lecture02</i>	p. 12		
<b>strncmp()</b>			
<i>lecture02</i>	p. 12,13		
<b>strncpy()</b>			
<i>lecture02</i>	p. 12		
Stroustrup, Bjarne			
<i>lecture08</i>	p. 11-14		
<b>strchr()</b>			
<i>lecture02</i>	p. 13		
<b>strsep()</b>			
<i>lecture02</i>	p. 14		
<b>strstr()</b>			
<i>lecture02</i>	p. 13		
<b>strtod()</b>			
<i>lecture03</i>	p. 1		
<b>strtok()</b>			

## T

Tail pointer		<i>lecture06</i>	p. 7,8
Thomson (Ken)		<i>lecture01</i>	p. 7
Thomson, Ken		<i>lecture01</i>	p. 6
Threads		<i>lecture09</i>	p. 16
<b>throw</b>		<i>lecture08</i>	p. 16
Time functions		<i>lecture03</i>	p. 2,20,21
<b>time()</b>			

<i>lecture03</i>	p. 2,21
<b>time.h</b>	
<i>lecture03</i>	p. 21
<i>lecture03</i>	p. 2,21
<b>timegm()</b>	
<i>lecture03</i>	p. 21
<b>time_t</b>	
<i>lecture03</i>	p. 21
<i>lecture03</i>	p. 2
Tokenizing	
<i>lecture02</i>	p. 13,14
<b>tolower()</b>	
<i>lecture02</i>	p. 11
Tools	
<i>lecture09</i>	p. 1
<b>toupper()</b>	
<i>lecture02</i>	p. 11
Towers of Hanoi	
<i>lecture05</i>	p. 15
Tree	
<i>lecture06</i>	p. 13,20-23
<i>lecture07</i>	p. 1,2,4
<b>try</b>	
<i>lecture08</i>	p. 16
<b>typedef</b>	
<i>lecture03</i>	p. 17
<i>lecture09</i>	p. 2

## U

Unicode	
<i>lecture02</i>	p. 15,17,18
<b>union</b>	
<i>lecture03</i>	p. 23
<i>lecture03</i>	p. 24
<b>unistd.h</b>	
<i>lecture09</i>	p. 8
UNIX	
<i>lecture01</i>	p. 6
Unix	
<i>lecture01</i>	p. 7
Unix pipe	
<i>lecture02</i>	p. 9
<b>unlink()</b>	

<i>lecture04</i>	p. 2
<b>unsigned</b>	
<i>lecture01</i>	p. 11,12
UTF-16	
<i>lecture02</i>	p. 17
UTF-32	
<i>lecture02</i>	p. 17
UTF-8	
<i>lecture02</i>	p. 15,18

## V

Variable declaration	
<i>lecture01</i>	p. 9
Variable name	
<i>lecture01</i>	p. 9
Variable number of parameters	
<i>lecture02</i>	p. 6
<b>vector</b>	
<i>lecture08</i>	p. 19,20
Virtual machine	
<i>lecture09</i>	p. 7
Visual C++	
<i>lecture09</i>	p. 22
Visual Studio	
<i>lecture01</i>	p. 7
<b>void</b>	
<i>lecture04</i>	p. 13
<b>void*</b>	
<i>lecture04</i>	p. 18
Von Neumann (John)	
<i>lecture01</i>	p. 8
Von Neumann, John	
<i>lecture04</i>	p. 9

## W

Walking a binary tree	
<i>lecture06</i>	p. 16
Wall gcc flag	
<i>lecture09</i>	p. 22
<b>wchar</b>	
<i>lecture02</i>	p. 14,15

## **while**

*lecture02* p. 1

Wide char

*lecture02* p. 14,15

## **wine**

*lecture09* p. 7

# X

Xcode

*lecture01* p. 7

*lecture09* p. 22

## **XML**

*lecture04* p. 3

# Z

## **ZIP**

*lecture04* p. 3