

!= (Logical operator not)	
<i>lecture01</i>	p. 23
" (Double quote)	
<i>lecture01</i>	p. 15
#define	
<i>lecture01</i>	p. 16,18
#include	
<i>lecture01</i>	p. 16,20
<i>lecture02</i>	p. 7
& (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
\0	
<i>lecture01</i>	p. 14,15
<i>lecture02</i>	p. 10-12
\n	
<i>lecture02</i>	p. 10
((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
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
argc	
<i>lecture02</i>	p. 2
<i>lecture03</i>	p. 14,15
<i>lecture03</i>	p. 13
argv[]	
<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
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

assert

<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
atof()	
<i>lecture03</i>	p. 1
atoi()	
<i>lecture03</i>	p. 1
atol()	
<i>lecture03</i>	p. 1

B

Bell Labs	
<i>lecture01</i>	p. 6
Bell labs	
<i>lecture01</i>	p. 7
Binary file	
<i>lecture04</i>	p. 2
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
break	
<i>lecture01</i>	p. 24
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
C11	
<i>lecture01</i>	p. 7
C89	
<i>lecture01</i>	p. 7
C99	
<i>lecture01</i>	p. 7
Calling functions	
<i>lecture04</i>	p. 8
calloc()	
<i>lecture04</i>	p. 18
case	
<i>lecture01</i>	p. 24
Case	
<i>lecture02</i>	p. 11
Case insensitive comparison	
<i>lecture02</i>	p. 13
CFLAGS	
<i>lecture04</i>	p. 5
Changing case	
<i>lecture02</i>	p. 11
char	
<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
CJK	

<i>lecture02</i>	p. 15
Classification of characters	
<i>lecture02</i>	p. 10,11
Code	
<i>lecture01</i>	p. 8
codepoint	
<i>lecture02</i>	p. 15
<i>lecture02</i>	p. 15
Collections	
<i>lecture05</i>	p. 16
Command-line parameters	
<i>lecture02</i>	p. 2
<i>lecture03</i>	p. 13-15
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
Constants	
<i>lecture01</i>	p. 16,18
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
Craftsmanship	
<i>lecture01</i>	p. 5
ctime()	
<i>lecture03</i>	p. 2,3,21
ctype.h	
<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

Data	
<i>lecture01</i>	p. 8
<i>lecture05</i>	p. 16,17
Data structure	
<i>lecture05</i>	p. 17
Data structures	
<i>lecture05</i>	p. 16,17,21,22
Data types	
<i>lecture01</i>	p. 10-12
Declaration	
<i>lecture05</i>	p. 1
Declaration of pointer	
<i>lecture03</i>	p. 5,6
Declaration of variable	
<i>lecture01</i>	p. 9
Deleting a file	
<i>lecture04</i>	p. 2
Dereferencing	
<i>lecture03</i>	p. 7,8,19,20
Direct access	
<i>lecture04</i>	p. 2
Directory operations	
<i>lecture04</i>	p. 3
dirent.h	
<i>lecture04</i>	p. 3
Distribution	
<i>lecture03</i>	p. 3
do ... while	
<i>lecture02</i>	p. 1
Dot reference to structure field	
<i>lecture03</i>	p. 16,20
double	
<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
Dumping a binary file	

<i>lecture04</i>	p. 3
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

EDP	
<i>lecture05</i>	p. 16
Electronic Data Processing	
<i>lecture05</i>	p. 16
else	
<i>lecture01</i>	p. 21,23
else if	
<i>lecture01</i>	p. 23
Encoding	
<i>lecture01</i>	p. 9
<i>lecture02</i>	p. 15
End-of-string marker	
<i>lecture01</i>	p. 14,15
EOF	
<i>lecture02</i>	p. 9,10
Epoch	
<i>lecture03</i>	p. 2
errno	
<i>lecture03</i>	p. 1
errno.h	
<i>lecture03</i>	p. 1
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
Exams	

<i>lecture01</i>	p. 2-4
Exception	
<i>lecture02</i>	p. 4,5
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
extern	
<i>lecture04</i>	p. 7
<i>lecture04</i>	p. 7

F

Factorial	
<i>lecture05</i>	p. 14
fclose()	
<i>lecture03</i>	p. 25
fepf()	
<i>lecture04</i>	p. 1
ferror()	
<i>lecture04</i>	p. 1
fgetc()	
<i>lecture02</i>	p. 9
<i>lecture04</i>	p. 1
fgets()	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 10
<i>lecture04</i>	p. 14
fgets():Return value	
<i>lecture03</i>	p. 1
FILE	
<i>lecture03</i>	p. 26
FILE *	
<i>lecture03</i>	p. 25
Files	
<i>lecture03</i>	p. 24-26
Final exam	
<i>lecture01</i>	p. 3
float	
<i>lecture01</i>	p. 12

flock()	
<i>lecture04</i>	p. 2
Flow control	
<i>lecture01</i>	p. 21,23,24
<i>lecture02</i>	p. 1
fopen()	
<i>lecture03</i>	p. 25,26
for	
<i>lecture02</i>	p. 1
Formatted input and output	
<i>lecture02</i>	p. 10
fprint()	
<i>lecture03</i>	p. 26
fprintf()	
<i>lecture02</i>	p. 10
fputc()	
<i>lecture02</i>	p. 9
<i>lecture04</i>	p. 1
fputs()	
<i>lecture02</i>	p. 10
<i>lecture03</i>	p. 26
fread()	
<i>lecture04</i>	p. 1
free()	
<i>lecture04</i>	p. 19
<i>lecture04</i>	p. 18,21-23
fseek()	
<i>lecture04</i>	p. 2
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 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
fwrite()	
<i>lecture04</i>	p. 1

G

Garbage collector	
<i>lecture04</i>	p. 19
gcc	
<i>lecture01</i>	p. 17
<i>lecture01</i>	p. 7
gcd()	
<i>lecture04</i>	p. 8,9
getchar()	
<i>lecture02</i>	p. 9
getopt()	
<i>lecture03</i>	p. 15
gets()	
<i>lecture02</i>	p. 10
Global variable	
<i>lecture03</i>	p. 1
<i>lecture04</i>	p. 15
gmtime()	
<i>lecture03</i>	p. 21
Grades	
<i>lecture01</i>	p. 4

H

Hanoi (towers of)	
<i>lecture05</i>	p. 15
head	
<i>lecture04</i>	p. 3
Header file	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 7
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	
<i>lecture01</i>	p. 5
I	
if	
<i>lecture01</i>	p. 21,23
Information	
<i>lecture05</i>	p. 16,17
Information Technology	
<i>lecture05</i>	p. 16
Initialization of pointer	
<i>lecture03</i>	p. 7,8
Initialization of structure	
<i>lecture03</i>	p. 16
Input/Output	
<i>lecture02</i>	p. 9,10
int	
<i>lecture01</i>	p. 11
integer operations	
<i>lecture01</i>	p. 11
isalnum()	
<i>lecture02</i>	p. 11
isalpha()	
<i>lecture02</i>	p. 11
isdigit()	
<i>lecture02</i>	p. 11
islower()	
<i>lecture02</i>	p. 11
ISO	
<i>lecture02</i>	p. 16
isprint()	
<i>lecture02</i>	p. 11
ispunct()	
<i>lecture02</i>	p. 11
isspace()	
<i>lecture02</i>	p. 11
isupper()	
<i>lecture02</i>	p. 11
IT	

<i>lecture05</i>	p. 16
------------------	-------

J

Java vs C	
<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
java vs C	
<i>lecture04</i>	p. 17
Java vs C	
<i>lecture05</i>	p. 1,2,16

K

K&R	
<i>lecture01</i>	p. 6
Keringhan (Brian)	
<i>lecture01</i>	p. 6

L

Lab2 hints	
<i>lecture05</i>	p. 3-5
Labs	
<i>lecture01</i>	p. 3,4
ld	
<i>lecture01</i>	p. 20
Library file	
<i>lecture04</i>	p. 6
Linked list	
<i>lecture05</i>	p. 22
Linker	
<i>lecture01</i>	p. 16,17,19,20
<i>lecture04</i>	p. 7-9
Linux	
<i>lecture01</i>	p. 2
localtime()	
<i>lecture03</i>	p. 21
Locking a file	
<i>lecture04</i>	p. 2

Logical operators
lecture01 p. 23

long
lecture01 p. 11
lecture01 p. 11

Loop
lecture02 p. 1

M

main()
lecture01 p. 16

make
lecture01 p. 7
lecture04 p. 5,6
lecture04 p. 4-6

Makefile
lecture04 p. 5,6

malloc()
lecture04 p. 18-20
lecture05 p. 20

man
lecture02 p. 5

Marker (end-of-string)
lecture01 p. 14,15

Mathematical functions
lecture01 p. 19,20

Mathematical functions:Compiler
lecture01 p. 19

Mathematical Induction
lecture05 p. 10

Mathematical induction
lecture05 p. 8-10

Maurolico, Francisco
lecture05 p. 9

memory
lecture01 p. 8
lecture01 p. 8

Memory address
lecture01 p. 9,10

Memory leak
lecture04 p. 22

Midcourse exam
lecture01 p. 3

mktime()
lecture03 p. 21

Multi-threading
lecture04 p. 15,16

Multidimensional array
lecture03 p. 13

N

Name of variable
lecture01 p. 9

Naming a structure
lecture03 p. 16,17

Nesting functions
lecture02 p. 6

Node
lecture05 p. 21,22

Normal distribution
lecture03 p. 3

Not (logical operator)
lecture01 p. 23

NULL
lecture02 p. 10,13
lecture03 p. 1,7

O

od
lecture04 p. 3

Or (logical operator)
lecture01 p. 23

Order
lecture05 p. 20,21

Overflow
lecture02 p. 12

Overloading
lecture02 p. 5

P

Pascal, Blaise
lecture05 p. 9

perror()

<i>lecture03</i>	p. 1
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 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
pptx	
<i>lecture04</i>	p. 3
Preprocessor	
<i>lecture01</i>	p. 16-18,20
printf()	
<i>lecture02</i>	p. 8,10
Prototype (function)	
<i>lecture01</i>	p. 16
Prototype (functions)	
<i>lecture02</i>	p. 7
putchar()	
<i>lecture02</i>	p. 9
puts()	
<i>lecture02</i>	p. 10

Q

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

R

Radix	
<i>lecture01</i>	p. 12
random()	
<i>lecture03</i>	p. 2,3
Reading ZIP or XML	
<i>lecture04</i>	p. 3
realloc()	
<i>lecture04</i>	p. 18
<i>lecture05</i>	p. 19,20
Recursion	
<i>lecture05</i>	p. 10-15
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
Rounding error	
<i>lecture01</i>	p. 12

S

scanf()	
<i>lecture01</i>	p. 16
<i>lecture02</i>	p. 3,4,10
<i>lecture04</i>	p. 14
Schedule	
<i>lecture01</i>	p. 1
Semi-colon	
<i>lecture01</i>	p. 15

setlocale		<i>lecture02</i>	p. 12
<i>lecture02</i>	p. 15	strchr()	
setlocale()		<i>lecture02</i>	p. 13
<i>lecture03</i>	p. 3	strcmp()	
Shared library		<i>lecture02</i>	p. 12,13
<i>lecture04</i>	p. 7	strcpy()	
short		<i>lecture02</i>	p. 12
<i>lecture01</i>	p. 11	strdup()	
signed		<i>lecture04</i>	p. 18
<i>lecture01</i>	p. 11,12	<i>lecture05</i>	p. 18
Single quote		Stream	
<i>lecture01</i>	p. 15	<i>lecture02</i>	p. 9
sizeof()		Stream redirection	
<i>lecture03</i>	p. 6,13	<i>lecture03</i>	p. 24
Sorting		strerror()	
<i>lecture05</i>	p. 6-8,11-14,19	<i>lecture03</i>	p. 1
sscanf()		String	
<i>lecture01</i>	p. 16	<i>lecture01</i>	p. 10,14,15
Stack		String array	
<i>lecture01</i>	p. 8	<i>lecture03</i>	p. 12,13
<i>lecture04</i>	p. 9-12	String comparison	
static		<i>lecture02</i>	p. 12,13
<i>lecture04</i>	p. 7,16	String conversion to number	
Static variable		<i>lecture03</i>	p. 1
<i>lecture04</i>	p. 16	String declaration	
stderr		<i>lecture03</i>	p. 11
<i>lecture02</i>	p. 9	String search	
<i>lecture02</i>	p. 9	<i>lecture02</i>	p. 13
<i>lecture03</i>	p. 1	string.h	
stdin		<i>lecture02</i>	p. 11-13
<i>lecture02</i>	p. 9	<i>lecture04</i>	p. 18
<i>lecture02</i>	p. 9,10	Strings	
<i>lecture03</i>	p. 24,25	<i>lecture02</i>	p. 11-13
stdio.h		<i>lecture03</i>	p. 1
<i>lecture03</i>	p. 25	strlen()	
stdlib.h		<i>lecture02</i>	p. 11
<i>lecture03</i>	p. 1	strncasecmp()	
<i>lecture04</i>	p. 18	<i>lecture02</i>	p. 13
stdout		strncat()	
<i>lecture02</i>	p. 9,10	<i>lecture02</i>	p. 12
<i>lecture02</i>	p. 9,10	strncmp()	
<i>lecture03</i>	p. 24,25	<i>lecture02</i>	p. 12,13
strcasecmp()		strncpy()	
<i>lecture02</i>	p. 13	<i>lecture02</i>	p. 12
strcat()		strrchr()	

<i>lecture02</i>	p. 13
strsep()	
<i>lecture02</i>	p. 14
strstr()	
<i>lecture02</i>	p. 13
strtod()	
<i>lecture03</i>	p. 1
strtok()	
<i>lecture02</i>	p. 13,14
strtol()	
<i>lecture03</i>	p. 1
Struct	
<i>lecture03</i>	p. 20
struct	
<i>lecture03</i>	p. 15,17,20,23
<i>lecture03</i>	p. 16-20
<i>lecture05</i>	p. 17
struct tm	
<i>lecture03</i>	p. 21
Structure alignment	
<i>lecture03</i>	p. 18
Structure and pointer	
<i>lecture03</i>	p. 19,20
Structure initialization	
<i>lecture03</i>	p. 16
Structure naming	
<i>lecture03</i>	p. 16,17
Structures	
<i>lecture03</i>	p. 15-20
switch	
<i>lecture01</i>	p. 24

T

Thomson (Ken)	
<i>lecture01</i>	p. 7
Thomson, Ken	
<i>lecture01</i>	p. 6
Time functions	
<i>lecture03</i>	p. 2,20,21
time()	
<i>lecture03</i>	p. 2,21
time.h	
<i>lecture03</i>	p. 21

<i>lecture03</i>	p. 2,21
timegm()	
<i>lecture03</i>	p. 21
time_t	
<i>lecture03</i>	p. 21
<i>lecture03</i>	p. 2
Tokenizing	
<i>lecture02</i>	p. 13,14
tolower()	
<i>lecture02</i>	p. 11
toupper()	
<i>lecture02</i>	p. 11
Towers of Hanoi	
<i>lecture05</i>	p. 15
typedef	
<i>lecture03</i>	p. 17

U

Unicode	
<i>lecture02</i>	p. 15,17,18
union	
<i>lecture03</i>	p. 23
<i>lecture03</i>	p. 24
UNIX	
<i>lecture01</i>	p. 6
Unix	
<i>lecture01</i>	p. 7
Unix pipe	
<i>lecture02</i>	p. 9
unlink()	
<i>lecture04</i>	p. 2
unsigned	
<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
Visual Studio	
<i>lecture01</i>	p. 7
void	
<i>lecture04</i>	p. 13
void*	
<i>lecture04</i>	p. 18
Von Neumann (John)	
<i>lecture01</i>	p. 8
Von Neumann, John	
<i>lecture04</i>	p. 9

W

wchar	
<i>lecture02</i>	p. 14,15
while	
<i>lecture02</i>	p. 1
Wide char	
<i>lecture02</i>	p. 14,15

X

Xcode	
<i>lecture01</i>	p. 7
XML	
<i>lecture04</i>	p. 3

Z

ZIP	
<i>lecture04</i>	p. 3