



UNIVERSITÀ DI PARMA



Informatica e Laboratorio di Programmazione

*The true sign of intelligence is not knowledge
but imagination.*

Albert Einstein

- $\sim \frac{2}{3}$ in Aula (slide + esempi)
- $\sim \frac{1}{3}$ in Laboratorio (programmazione)
- Registrazione lezioni (tentativo)
- Slide ed esempi su elly2022.dia.unipr.it
 - Caveat: farò modifiche!

- Lunedì 13:30-15:30 (Lab. E c/o Q02)
- Mercoledì 10:30-12:30 (Aula F c/o sede didattica ing.)
- Giovedì 8:30-10:30 (Aula F c/o sede didattica ing.)

- Modifiche

- 03/10 → 06/10 16:30-18:30
- 10/10 → 13/10 16:30-18:30
- 24/10 → 27/10 16:30-18:30

- Sospensioni

- 26/09 → Elezioni (recupero da stabilire)

- Si assume che gli studenti abbiano
 - Concetti base Tecnologie Informazione
 - Uso computer e gestione file
 - Saper leggere e scrivere....

- Parte teorica
 - aula
- Parte pratica
 - aula + laboratori
- Problem solving
 - aula + laboratori

- Parte teorica (aula)
- Informazioni base necessarie per capire cosa significa e come approcciarsi alla programmazione
 - Architettura di base di un elaboratore
 - Algoritmi
 - Rappresentazione numeri
 - Architettura base calcolatore

- Parte pratica (aula + laboratori)
- Focus sul linguaggio usato per le attività di programmazione
 - Basi del C
 - Variabili
 - Strutture di controllo
 - Espressioni
 - Array, stringhe e puntatori
 - Funzioni
 - I/O
- Slide + esempi

- Accesso laboratori
- Tramite username istituzionale:
 - nome.cognomeXY@studenti.unipr.it
- Oppure account generici:
 - Utente: nome macchina, Password: noncopiare
- Possibile uso proprio portatile
 - Non all'esame!

- Problem solving (aula + laboratorio)
- Analisi di vecchi testi di esame e prove pratiche
 - live

- Prova pratica in laboratorio
 - Discussione elaborato
 - ~2 ore
 - C
 - Modulare
- Valutazione
 - Deve compilare
 - Essere eseguibile
 - Non va “tentato”

- Correzioni online
 - www.ce.unipr.it/didattica/informatica

Correzioni di Informatica & Laboratorio di Programmazione

Da queste pagine potete accedere alle correzioni online delle prove pratiche di Informatica & Laboratorio di Programmazione

Elenco prove pratiche:

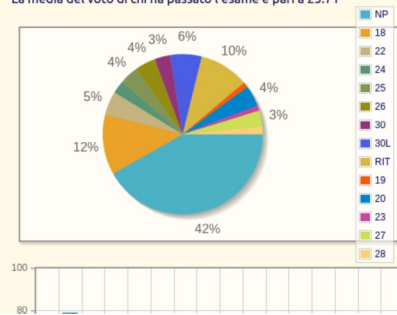
- 12 gennaio 2021 *Elenco telefonico, lettura file tipo CSV e memorizzazione in struttura dati allocata dinamicamente, ricerca in struttura dati*
- 28 gennaio 2021 *Dati comuni italiani, lettura file CSV in struttura dati allocata dinamicamente, ricerca in dati letti*
- 16 febbraio 2021 *Cambi storici valute, lettura file CSV e memorizzazione in struttura allocata dinamicamente, ricerca in struttura dati*
- 9 aprile 2021 *Voti carriera studente, lettura file con numero campi per riga variabile e allocazione dinamica memoria*
- 8 giugno 2021 *Analisi file log web server, memorizzazione dati con allocazione dinamica memoria e manipolazione stringhe*
- 13 luglio 2021 *Lettura parole da file, allocazione dinamica lettura stringa, analisi lunghezze stringhe*
- 20 dicembre 2021 *Enigmistica intarsio, lettura stringhe da file e relativa manipolazione, struttura a stack*
- 28 gennaio 2022 *Nested Set Model, memorizzazione dati da file CSV, ricerca nella struttura dati*
- 7 febbraio 2022 *Gioco impiccato, memorizzazione file dizionario, estrazione stringa casuale, ricerca caratteri*
- 17 febbraio 2022 *Anagrammi con frequenza uso lettere, manipolazione stringhe, ricerca in file*
- 11 aprile 2022 *Codice EAN-13, memorizzazione dati file, manipolazione stringhe e ricerca in dati*
- 8 giugno 2022 *Lettura immagine PGM, calcolo differenze tra valori adiacenti vettore, salvataggio dati*
- 13 luglio 2022 *Simulazione gioco carte, gestione array e rand()*



Statistiche dei risultati dall'A.A. 2021-2022

Hanno passato la prova: 66 su 132 con un tasso pari a 50.00% oppure pari a 44.90% considerando anche i ritirati

La media del voto di chi ha passato l'esame è pari a 23.71



- Non prendere sottogamba
- Chi non segue esercitazioni → risultato negativo
- Qualche numero (aprile 2022 su coorte 2021)
 - Superato esame 38%
 - Non superato 24,1%
 - Mai tentato 37,2%
 - n. Tentativi medi 1,7
 - Voto medio 23,5

- Martedì pomeriggio (tentativo) > 16:00
- Pal. 1 dei Cubi (sede scientifica ing.), primo piano



- Un buon manuale di C
 - K.N. King, *C Programming: A Modern Approach (2nd Edition)*, W W Norton & Co
 - Bellini Guidi, *Linguaggio C*, Mc Graw Hill
 - B.W. Kernighan e D. Ritchie, *The C Programming Language: ANSI C Version*, Pearson College
 - Darnell Margolis, *C manuale di programmazione*, Mc Graw Hill



<i>Returns</i>	c (the character written). If a write error occurs, putchar sets the stream's error indicator and returns EOF.	7.3, 22.4
puts	<i>Write String</i> <code>int puts(const char *s);</code> Writes the string pointed to by s to the stdout stream, then writes a new-line character.	<stdio.h>
<i>Returns</i>	A nonnegative value if successful. Returns EOF if a write error occurs.	13.3, 22.5
putwc	<i>Write Wide Character to File (C99)</i> <code>wint_t putwc(wchar_t c, FILE *stream);</code> Wide-character version of putc.	<wchar.h> 25.5
putwchar	<i>Write Wide Character (C99)</i> <code>wint_t putwchar(wchar_t c);</code> Wide-character version of putchar.	<wchar.h> 25.5
qsort	<i>Sort Array</i> <code>void qsort(void *base, size_t nmemb, size_t size, int (*compar)(const void *, const void *));</code> Sorts the array pointed to by base. The array has nmemb elements, each size bytes long. compar is a pointer to a comparison function. When passed pointers to two array elements, the comparison function must return a negative, zero, or positive integer, depending on whether the first array element is less than, equal to, or greater than the second.	<stdlib.h> 17.7, 26.2
raise	<i>Raise Signal</i> <code>int raise(int sig);</code> Raises the signal whose number is sig.	<signal.h>
<i>Returns</i>	Zero if successful, nonzero otherwise.	24.3
rand	<i>Generate Pseudo-Random Number</i> <code>int rand(void);</code>	<stdlib.h>
<i>Returns</i>	A pseudo-random integer between 0 and RAND_MAX (inclusive).	26.2
realloc	<i>Resize Memory Block</i> <code>void *realloc(void *ptr, size_t size);</code> ptr is assumed to point to a block of memory previously obtained from calloc, malloc, or realloc. realloc allocates a block of size bytes, copying the contents of the old block if necessary.	<stdlib.h>
<i>Returns</i>	A pointer to the beginning of the new memory block. Returns a null pointer if a block of the requested size can't be allocated.	17.3



A.14.2 Pseudo-Random Number Generator Functions

The *rand()* and *srand()* functions enable you to generate pseudo-random numbers.

A.14.2.1 The *rand()* Function

```
#include <stdlib.h>
int rand( void );
```

The *rand()* function returns an integer in the range 0 through *RAND_MAX*. Successive calls to *rand()* should produce different integers. However, the sequence of random numbers could be the same for each program execution unless you use a different seed value via the *srand()* function.

A.14.2.2 The *srand()* Function

```
#include <stdlib.h>
void srand( unsigned int seed );
```

The *srand()* function uses the argument as a seed for a new sequence of pseudo-random numbers to be returned by subsequent calls to *rand()*. If *srand()* is invoked with the same seed value, the sequence of generated numbers will be the same. The default seed value is 1.

A.14.3 Memory Management Functions

The memory management functions enable you to allocate and deallocate memory dynamically. See Chapter 7 for more information about these functions.

A.14.3.1 The *calloc()* Function

```
#include <stdlib.h>
void *calloc( size_t nmemb, size_t size );
```

The *calloc()* function allocates contiguous space for *nmemb* objects, each of which has a length in bytes specified by *size*. All bits in the allocated space are initialized to zero. *calloc()* returns a pointer to the first byte of the allocated space. If the space cannot be allocated, or if *nelem* or *size* is zero, *calloc()* returns a null pointer.

- Massimo Bertozzi
- bertozzi@ce.unipr.it
- 0521 90 5845



UNIVERSITÀ DI PARMA

Informatica e Laboratorio di Programmazione



nec scire fas est omnia

Quinto Orazio Flacco, Odi