Utilizzo delle librerie

Prof. Salvatore Venticinque
Prof. Emanuele Bellini

Utilizzare le librerie

- Librerie standard del linguaggio:
  - Studiare i prototipi delle funzioni
  - Includere il file header
  - Passare correttamente i parametri
- Librerie di terze parti
  - Installare la libreria
  - Studiare li prototipi delle funzioni
  - Includere il file header
  - Passare correttamente i parametri

In ogni caso: imparare a utilizzare la documentazione.

Librerie standard

- Nel sistema sono già presenti
  - I file oggetto della libreria
    - *.o,*.a (archivi di *.o)
    - *.so (linux), *.dll (windows)
  - I file header contenenti i prototipi

Librerie standard

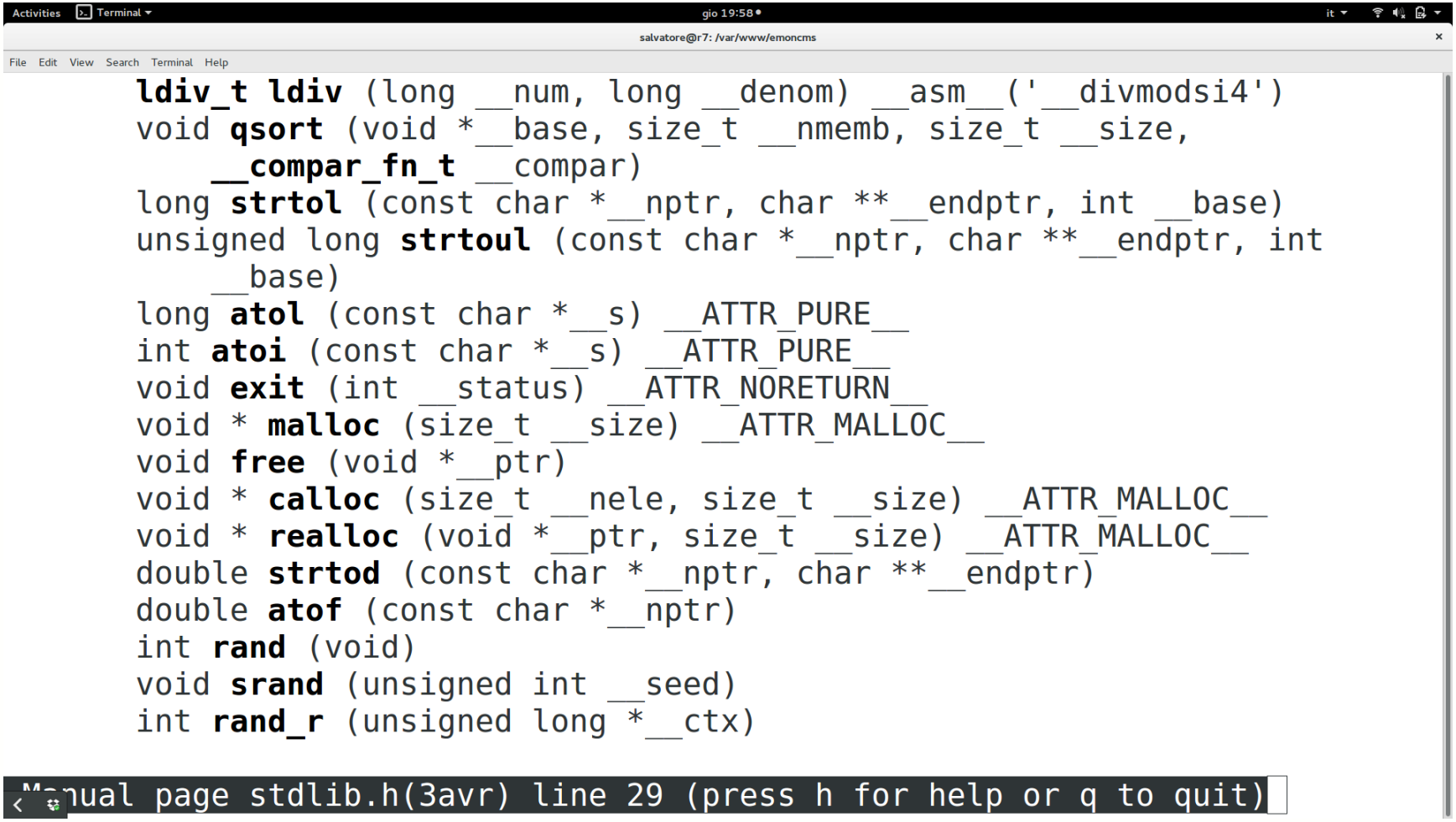
File header	Area di riferimento
`<assert.h>`	Diagnostica.
`<ctype.h>`	Controllo e conversione caratteri. Per esempio: `isdigit(c)` ritorna un valore diverso da zero se `c` è una cifra decimale; analogamente operano `isalpha(c)`, `isspace(c)`, `isupper(c)` ecc.
`<Errno.h>`	Segnalazioni di errore.
`<float.h>`	Floating point.
`<limists.h>`	Definisce alcune costanti come `INT_MAX` e `INT_MIN` che contengono rispettivamente il massimo e il minimo valore rappresentabile con un `int`.
`<locate.h>`	Inizializzazione dei parametri locali.

Librerie standard

`<math.h>`	Funzioni matematiche in doppia precisione.
`<setjmp.h>`	Salti non locali. Contiene la dichiarazione di funzioni che permettono di alterare l'esecuzione della normale sequenza di chiamata e uscita di una funzione, per esempio per obbligare a un ritorno immediato da una chiamata di funzione profondamente annidata.
`<signal.h>`	Gestione segnali. Contiene la dichiarazione di funzioni per la gestione di condizioni di eccezione che si verificano durante l'esecuzione, come l'arrivo di un segnale di interrupt da una sorgente esterna, oppure un errore nell'esecuzione.
`<stdarg.h>`	Gestione lista di argomenti variabili in numero e tipo. Contiene funzioni e/o macro che permettono di scandire tali liste, quindi può essere utile a sua volta per la realizzazione di funzioni che accettano un numero variabile di parametri.
`<stddef.h>`	Definizioni standard. Per esempio contiene la definizione di `ptrdiff_t` in grado di contenere la differenza, con segno, tra due puntatori e `size_t` il tipo (intero privo di segno) prodotto dalla funzione `sizeof`.
`<stdio.h>`	Input e Output. Funzioni quali `printf` e `scanf`.
`<stdlib.h>`	Utilità generale. Per esempio le funzioni per la conversione dei numeri, come `atof`, che trasforma una stringa in un `double`, o `rand` che ritorna un numero pseudo causale.
`<string.h>`	Gestione di stringhe. Funzioni quali **`strcpy`**, che consente di copiare una stringa su un altra e **`strcat`** che concatena due stringhe.
`<time.h>`	Gestione della data e dell'ora. Per esempio la funzione `time` che ritorna l'ora corrente.

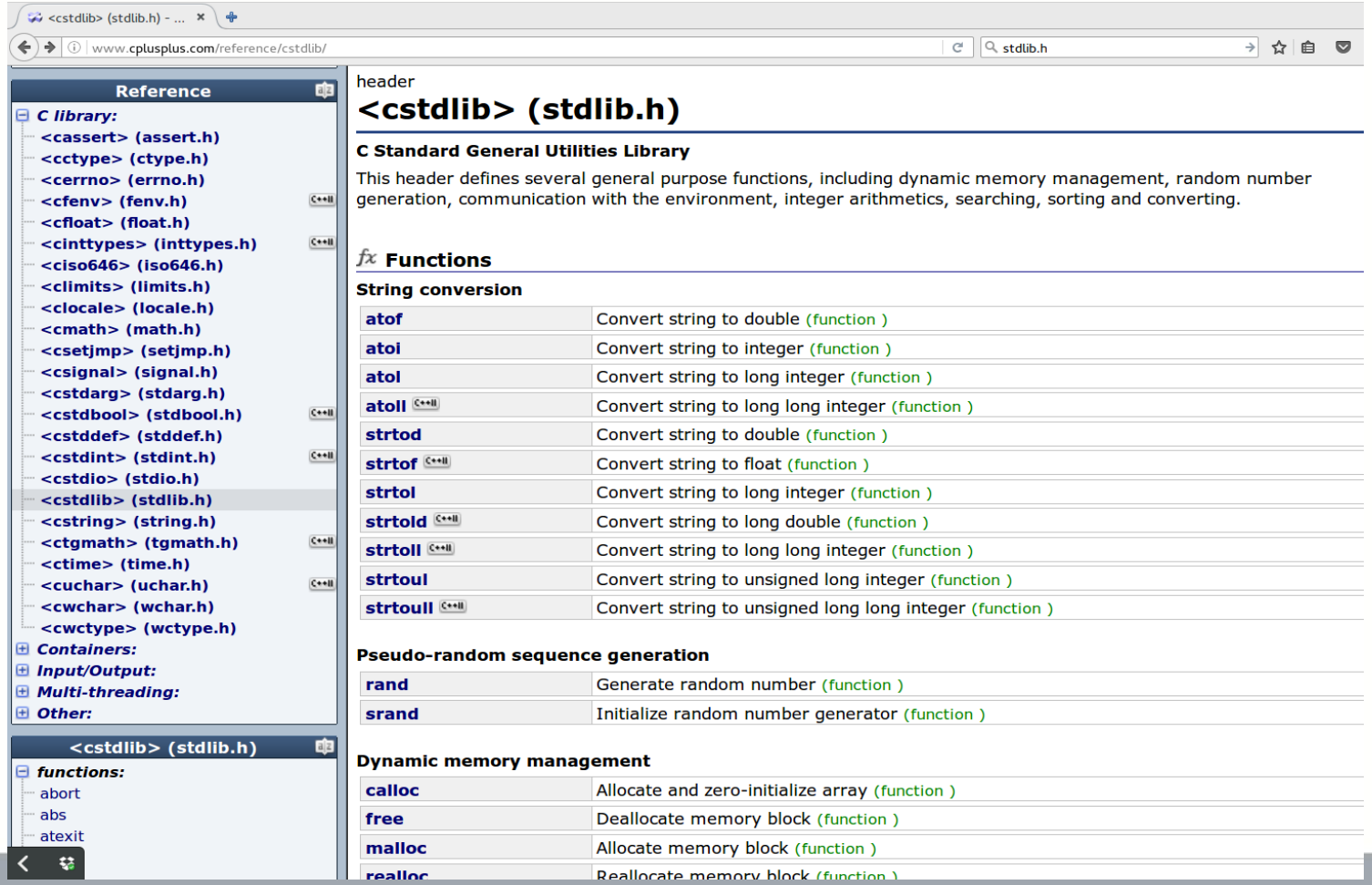
Librerie standard

- In linux: man stdlib.h



stdlib.h

- Da Internet: http://www.cplusplus.com/reference/cstdlib/



function

rand

<cstdlib>

```
int rand (void);
```

Generate random number

Returns a pseudo-random integral number in the range between 0 and RAND_MAX.

This number is generated by an algorithm that returns a sequence of apparently non-related numbers each time it is called. This algorithm uses a seed to generate the series, which should be initialized to some distinctive value using function srand.

RAND_MAX is a constant defined in <cstdlib>.

A typical way to generate trivial pseudo-random numbers in a determined range using rand is to use the modulo of the returned value by the range span and add the initial value of the range:

```
1 v1 = rand() % 100;         // v1 in the range 0 to 99
2 v2 = rand() % 100 + 1;     // v2 in the range 1 to 100
3 v3 = rand() % 30 + 1985;   // v3 in the range 1985-2014
```

Notice though that this modulo operation does not generate uniformly distributed random numbers in the span (since in most cases this operation makes lower numbers slightly more likely).

C++ supports a wide range of powerful tools to generate random and pseudo-random numbers (see <random> for more info).

Parameters

(none)

Return Value

An integer value between 0 and RAND_MAX.

Utilizzo di int rand()

Example

```c
/* rand example: guess the number */
#include <stdio.h>      /* printf, scanf, puts, NULL */
#include <stdlib.h>     /* srand, rand */
#include <time.h>       /* time */

int main ()
{
  int iSecret, iGuess;

  /* initialize random seed: */
  srand (time(NULL));

  /* generate secret number between 1 and 10: */
  iSecret = rand() % 10 + 1;

  do {
    printf ("Guess the number (1 to 10): ");
    scanf ("%d",&iGuess);
    if (iSecret<iGuess) puts ("The secret number is lower");
    else if (iSecret>iGuess) puts ("The secret number is higher");
  } while (iSecret!=iGuess);

  puts ("Congratulations!");
  return 0;
}
```

In this example, the random seed is initialized to a value representing the current time (calling `time`) to generate a different value every time the program is run.

Possible output:

```
Guess the number (1 to 10): 5
The secret number is higher
Guess the number (1 to 10): 8
The secret number is lower
Guess the number (1 to 10): 7
Congratulations!
```

Utilizzo di int rand()

int rand()

- È una funzione che appartiene alla libreria stdlib.h
- Ritorna un intero da una sequenza random compreso tra 0 e RAN_MAX
- Cosa succede se viene eseguito più volte il seguente programma?

```
int a =random();
printf("%d\n",a);
printf("%d\n",random());
```

Utilizzo di int rand()

```
// v1 in the range 0 to 99
v1 = rand() % 100;


// v2 in the range 1 to 100
v2 = rand() % 100 + 1;


 // v3 in the range 1985-2014
v3 = rand() % 30 + 1985;
```

Utilizzo di srand(time_t)

- void srand (unsigned int seed);
-
- la sequenza di numeri random.

Es:

```
srand(10);
int a =random();
printf(“%d\n”,a);
printf(“%d\n”,random());
```

Stamperà una nuova coppia di numeri random!!!

Utilizzo di srand()

```
#include <stdlib.h>
#include <time.h>

int main()
{
  srand(time(NULL));
  int a =random();
   printf(“%d\n”,a);
   printf(random());
}
```

math.h

- double cos (double x);
- double sin (double x);
- double tan (double x);
- [...]
- double cosh(double x);
- [...]
- double log2 (double x);
- float log2f (float x);
- double exp (double x);
-

log

double log(double d)

- Ritorna il logaritmo naturale di d
- Se d è negativo viene settata una variabile globale **errno**

- **in caso di errore erno è uguale ad un valore intero dichiarato come costante EDOM in errno.h**

Esempio

```
#include <stdio.h>
#include <math.h>
#include <errno.h>
int main()
{
   double d=log(123.4);
   if(errno==EDOM)
        printf("errore\n");
   else
       printf("%f\n",d);
    d=log(-123.4);
    if(errno==EDOM)
        printf("errore\n");
    else
        printf("%f\n",d);
}
```

Alcuni esercizi

- Realizzare un programma che consente all'utente di giocare contro il computer a carta,forbice,sasso.
- Realizzare un programma che stampa la figura del coseno in verticale a console.
- Realizzare il calcolo del pigreco come rapporto tra i punti random che cadono all'interno di un cerchio di raggio r e all'interno del quadrato di lato 2r in cui il cerchio è iscritto.