Laboratorio di Calcolo Canale Lp-P, Gruppo A
Esercitazione Valutata del 20/12/2021

Docenti: Prof. C. De Michele e Prof. A. Paiella

Per lo svolgimento di tale prova avete a disposizione 3 ore. Verranno valutati eventuali commenti e l'indentazione del codice che potranno costituire un bonus ulteriore nel punteggio finale. E' ammesso l'utilizzo degli appunti, del libro di testo e dei prontuari di python e linux. All'interno di ogni codice riportare un commento con il proprio nome, cognome e numero di matricola.

► **Prima parte:** Fate il login con username e password che vi sono stati assegnati per le precedenti esercitazioni. Ad esempio, se vi era stato assegnato il gruppo `011` dovrete usare come username `lccdm011` e come password `lcccdm011`.
Create quindi una cartella chiamata EX10_<NOME>_<COGNOME> che dovrà contenere tutto quello che produrrete durante questa esercitazione. Ad esempio se vi chiamate Luigi Rossi, dovete creare una cartella chiamata `EX10_LUIGI_ROSSI`. Lo scopo di questa esercitazione valutata è di scrivere un programma in C, chiamato `<nome>_<cognome>.c`, che calcoli il complemento a due di un numero binario di M bit, con $4 \leq M \leq 20$.
Il programma dovrà prima richiedere in input M come un intero e dovrà poi richiedere il numero binario costituito da M cifre come una stringa chiamata `numstr` (che sarà quindi costituita da M cifre binarie). Relativamente all'immissione del numero M, si dovrà controllare che questo sia un numero intero compreso tra 4 e 20, reiterando la richiesta di immissione nel caso che il controllo abbia esito negativo. Il programma, quindi, dovrà memorizzare il numero binario immesso in un array di M interi chiamato `numbin` (dove il primo elemento dell'array conterrà il bit meno significativo) e stampare su schermo il contenuto dell'array `numbin`. Infine, si dovrà calcolare il complemento a due del numero memorizzato nell'array `numbin` e lo si dovrà memorizzare in un'array di M interi chiamato `numcompl2` per poi stampare il contenuto di tale array su schermo. Come nel caso precedente, l'array `numcompl2` conterrà come primo elemento il bit meno significativo del numero.
In particolare il programma dovrà essere sviluppato secondo le seguenti specifiche:

1. Nel `main` si dovranno dichiarare due array costituiti da 20 interi chiamati `numbin` e `numcompl2`.

2. Si dovrà implementare una funzione chiamata `immettiM` che richiederà come argomento un puntatore ad intero e che dovrà richiedere in input il numero M controllando che sia nell'intervallo di valori richiesto. Il valore di M deve essere restituito alla funzione chiamante tramite il puntatore passato come argomento.

3. Si dovrà implementare una funzione chiamata `immettiNumero` che richiederà come argomento un puntatore a carattere e che dovrà richiedere in input una stringa che rappresenterà un numero binario positivo.

4. Si dovrà implementare una funzione chiamata `convertiStringa()` a cui si dovranno passare come argomenti la stringa `numstr`, l'array `numbin` ed il numero M. Tale funzione convertirà la stringa data in input nell'array di interi dove il primo elemento dell'array corrisponderà al bit meno significativo del numero binario immesso come stringa. In tale funzione si dovrà controllare che la stringa sia composta dai soli caratteri `'0'` o `'1'` e che questi siano in numero pari a M,

stampando un opportuno messaggio d'errore ed uscendo dal programma se la stringa non fosse conforme a tali specifiche.

5. Si dovrà implementare una funzione chiamata `calccompl2()` a cui si dovrà passare come argomento l'array `numbin`, l'array `numcompl2` ed il valore M, e che dovrà memorizzare nell'array `numcompl2` il complemento a due del numero contenuto in `numbin`. La conversione di un numero binario nel suo complemento a due operando sui singoli bit si ottiene invertendo prima tutti i bit e poi sommando 1.

6. Si dovrà implementare una funzione chiamata `stampanumero()` che prenderà come argomenti un stringa con del testo informativo da stampare (ad es. `"il numero in complemento a due è:"`), un puntatore ad intero da utilizzare per passare alla funzione l'array d'interi da stampare ed il numero M di bit di tale numero. Tale funzione dovrà stampare da sinistra a destra i bit dal più significativo (quello con indice M-1) al meno significativo (quello con indice 0) tramite un opportuno ciclo. Questa funzione dovrà essere utilizzata per stampare gli array `numbin` e `numcompl2`.

► **Seconda parte:** Modificare il programma scritto nella prima parte facendo in modo che, se $M = 0$, allora il programma memorizzi nell'array `numbin` un numero binario costituito da 20 bit casuali. Il complemento a due di tale numero generato casualmente e memorizzato nell'array `numcompl2` andrà poi salvato in un file chiamato `compl2.dat` (utilizzando le istruzioni `fopen()`, `fprintf()` e `fclose()`) nel seguente formato:

0	<cifra binaria meno significativa>
1	.
2	.
.	.
.	.
M-1	<cifra binaria più significativa>

► **Terza parte:** Creare uno script python chiamato `<nome>_<cognome>.py` che legga il file "compl2.dat" e ne grafichi il contenuto come un istogramma. Fare in modo che lo script stampi i nomi degli assi, una legenda ed un titolo. Lo script dovrà anche salvare il grafico su di un file chiamato "compl2.png"