	Politecnico di Milano Scuola di Ingegneria Industriale e dell'Informazione **FONDAMENTI DI INFORMATICA** Appello 21 Luglio 2018		COGNOME E NOME
	RIGA	COLONNA	CODICE PERSONA
			Spazio riservato ai docenti

- Il presente plico contiene 5 esercizi e **deve essere debitamente compilato con cognome e nome, codice persona**.
- Il tempo a disposizione è di 2 ore.
- Non separate questi fogli. Scrivete la soluzione solo sui fogli distribuiti, utilizzando il retro delle pagine in caso di necessità. Cancellate le parti di brutta con un tratto di penna.
- Ogni parte non cancellata a penna sarà considerata parte integrante della soluzione.
- È possibile scrivere a matita (e non occorre ricalcare al momento della consegna!).
- **È vietato utilizzare telefoni o pc**. Chi tenti di farlo vedrà annullata la sua prova.
- **Non è ammessa la consultazione di libri e appunti.**
- **Qualsiasi tentativo di comunicare con altri studenti comporta l'espulsione dall'aula**.
- È possibile ritirarsi senza penalità.
- **Non è possibile lasciare l'aula conservando il tema della prova in corso**.

Esercizio 1 (7 punti)

Implementare la seguente funzione:

```
int somma (int op1[], int op2[],  int sum[], int nbits),
```

dove `op1` e `op2` sono due vettori che contengono le cifre di due numeri binari in CPL2 (il primo elemento del vettore è la cifra più significativa, l'ultimo quella meno significativa); `nbits` è il numero di bit con cui `op1` e `op2` sono rappresentati; `sum` è l'indirizzo di un vettore che può ospitare `nbits` interi. La scopo della funzione è fare la somma in CPL2 su `nbits` di `op1` e `op2`, salvare il risultato nel vettore `sum` (mettendo in prima posizione la cifra più significativa e in ultima quella meno significativa), e restituire 1 se si è verificato un overflow, 0 altrimenti.

Soluzione

```
int somma (int op1[], int op2[], int sum[], int nbits)
{
  int i,carry=0;
  for (i=nbits-1; i>=0; i--)
  {
    int r = carry + op1[i] + op2[i];

    if (r==3)
    {
      carry = 1;
      sum[i] = 1;
    }
    if (r==2)
    {
      carry = 1;
      sum[i] = 0;
    }
    if (r==1 || r==0)
    {
      carry = 0;
      sum[i] = r;
    }
  }

  //calcolo condizione overflow
  return (op1[0] == op2[0]) && (op1[0] != sum[0]);
}
```

Esercizio 2 (8 punti)

Si consideri la seguente struttura dati per rappresentare una lista di numeri interi:

```
struct nodo
{
  int data;
  struct nodo *next;
};

typedef struct nodo *lista;
```

Si implementi la funzione `lista unici(lista a, lista b)`, che riceve in ingresso la testa di due liste (a e b) e restituisce la testa di una nuova lista che contiene solo gli elementi che sono presenti **soltanto** nella lista a oppure **soltanto** nella lista b.

Note. Gli elementi nella lista restituita possono comparire in qualsiasi ordine. È possibile utilizzare funzioni di supporto (ad esempio per inserire un elemento nella lista), parchè ne sia fornita l'implementazione. Si può assumere che le liste non contengano elementi ripetuti.

Soluzione

```
void inserisci_testa(lista *l, int data)
{
  struct nodo *temp = malloc(sizeof(struct nodo));
  temp->data = data;
  temp->next = *l;
  *l = temp;
}

int cerca(lista a, int data)
{
  if (a==NULL)
    return 0;
  else if (a->data == data)
    return 1;
  else
    return cerca(a->next, data);
}

lista unici(lista a, lista b)
{
  lista c=NULL, cur;

  cur = a;
  while (cur!=NULL)
  {
    if (cerca(b, cur->data)==0)
      inserisci_testa(&c, cur->data);
    cur = cur->next;
  }

  cur = b;
  while (cur!=NULL)
  {
    if (cerca(a, cur->data)==0)
      inserisci_testa(&c, cur->data);
    cur = cur->next;
  }

  return c;
}
```

Esercizio 3 (8 punti)

Si supponga di voler analizzare un file di testo nel quale ogni riga è un messaggio twitter rappresentato con la seguente struttura:

```
Questo è un esempio #fi #secondoappello #esercizio3
```

Dove `Questo è un esempio #fi #secondoappello #esercizio3` è il testo del messaggio, `fi`, `secondoappello` e `esercizio3` sono i 3 hashtag contenuti nel messaggio. Gli hasthag nel messaggio non possono essere più di 100, non possono avere più di 279 caratteri e non possono contenere spazi. Il testo del messaggio complessivamente non può contenere più di 280 caratteri (hashtag inclusi).

Si implementi la seguente funzione C:

```
int conta(char filename[], char hashtag[]),
```

che riceve in ingresso il nome (`filename`) di un file di testo contenente messaggi twitter, rappresentati nel formato descritto in precedenza, e una stringa contenente uno specifico hashtag; la funzione ha il compito di leggere tutti i messaggi twitter contenuti nel file `filename` e restituire il numero di messaggi che contengono l'hashtag passato come parametro di ingresso.

Nota. È possibile assumere che lo stesso hashtag non possa essere ripetuto più di una volta all'interno dello stesso messaggio twitter.

Soluzione

```
int conta(char filename[], char hashtag[])
{
  char str[282],htag[281];
  int i,j,trovato,n=0;
  FILE *in = fopen(filename,"r");

  while (fgets(str,282,in)!=NULL)
  {
    trovato = 0;
    for (i=0; i<strlen(str) && trovato==0;  i++)
    {

      if (str[i]=='#')
      {
        for (j=0; j+i+1 < strlen(str) && str[j+i+1]!=' '
                                      && str[j+i+1]!='\n'; j++)
          htag[j] = str[j+i+1];
        htag[j]='\0';

        if (strcmp(htag,hashtag)==0)
          trovato=1;
        i=i+j+1;
      }
      n = n + trovato;
    }
  }
  fclose(in);
  return n;
}
```

Esercizio 4 (4 punti)

Un sistema ha una memoria con le seguenti caratteristiche: memoria fisica di 64 Mbyte e memoria virtuale di 256 Mbyte con pagine di dimensione 16 Kbyte.

Rispondere alle seguenti domande giustificando le risposte:

A. Quale è la struttura dell'indirizzo virtuale e di quello fisico, e la lunghezza dei campi che li costituiscono?
B. Un consulente suggerisce di utilizzare uno swap file di dimensione non superiore a 192 Mbyte, ovvero pari alla differenza fra 256 Mbyte e 64 Mbyte, perché un file di dimensione maggiore sarebbe soltanto uno spreco di memoria. Sei d'accordo con questa affermazione? Giustificare la risposta.

Soluzione

A. 64 Mbyte di memoria fisica indirizzabile → 26 bit di indirizzo fisico
256 Mbyte di memoria virtuale indirizzabile → 28 bit di indirizzo virtuale
Pagina di 16Kbyte = 2^14 byte → 14 bit offset

Quindi si ha,

Indirizzo virtuale:	NPV: 14 bit, offset: 14 bit
Indirizzo fisico:	NPF: 12 bit, offset: 14 bit

B. L'affermazione non è corretta, dal momento che il file di swap viene utilizzato per memorizzare le pagine non residenti di tutti i processi e quindi potrebbe richiedere uno spazio complessivamente maggiore di 192 Mbyte.

Esercizio 5 (5 punti)

Si consideri il seguente programma in linguaggio macchina:

```
1.  LOAD= 0
2.  STORE 101
3.  READ
4.  STORE 102
5.  READ
6.  STORE 103
7.  MULT 103
8.  STORE 103
9.  READ
10. SUB 103
11. BNE 15
12. LOAD 101
13. ADD= 1
14. STORE 101
15. LOAD 102
16. SUB= 1
17. STORE 102
18. BG 5
19. LOAD 101
20. WRITE
21. END
```

Rispondere alle seguenti domande:

A) Ipotizzando che il nastro di ingresso contenga i valori {3,2,4,3,5,4,16}. Cosa viene scritto sul nastro di uscita?
B) Ipotizzando che il nastro di ingresso contenga i valori {4,4,4,3,6,5,25,6,12}. Cosa viene scritto sul nastro di uscita?
C) Spiegare sinteticamente cosa fa il programma in linguaggio macchina sopra riportato.

Soluzione

Il nastro di uscita conterrà 2 nel caso A e 1 nel caso B.
Il programma legge dal nastro di ingresso un intero N e, successivamente, N coppie di interi. Quindi scrive sul nastro di uscita il numero di coppie di interi lette in cui il primo intero è pari al quadrato del secondo intero.