word counter

Nei sistemi operativi Unix-like, il comando we consente di contare il numero di righe, parole, caratteri e byte di ciascun file o input standard e stampare il risultato.

Ad esempio il comando:

```
echo "ciao mondo\n" | wc
```

stamperà:

1 2 11

Il primo numero rappresenta il numero di righe, il secondo il numero di parole e il terzo il numero di caratteri.



Per avere una panoramica completa delle opzioni del comando we si può consultare la pagina del manuale digitando man we nel terminale.

Si vuole implementare in C una funzione **wc** che, data una stringa, restituisca il numero di righe, parole e caratteri totali.

Il prototipo della funzione è il seguente:

```
void wc(const char *string, Stats *counter);
```

dove:

• string è una stringa;

• counter è un puntatore ad una struttura Stats definita come segue:

```
typedef struct {
   int lines;
   int words;
   int chars;
} Stats;
```

La funzione deve calcolare il numero di righe, parole e caratteri della stringa string e memorizzarli nella struttura Stats puntata da counter.

Si assuma che:

- una riga è una sequenza di caratteri terminata da un carattere di nuova riga (\n);
- una parola è una sequenza di caratteri separata da spazi o da caratteri di nuova riga; "a
 b" sono due parole, "a\nb" sono due parole parola, "a,b" è una parola;

Se necessario, si possono aggiungere funzioni di supporto, ma è necessario che le funzioni richieste siano implementate esattamente con i prototipi forniti.

Soluzione (fare click per visualizzare)

Esistono diversi modi in C per implementare il codice richiesto, la soluzione proposta utilizza un ciclo while per iterare su tutti i caratteri della stringa e contare il numero di righe, parole e caratteri, utilizzando una variabile prev per tenere traccia del carattere precedente. La soluzione ha complessità O(n), dove n è la lunghezza della stringa. La soluzione proposta è la seguente:

```
void wc(const char *str, Stats *counter)
    // se la stringa è vuota non c'è nulla da contare
    if (*str == '\0')
        // dal momento che il contador è inizializzato a 0 non è necessario fare nulla
        return;
    }
    char prev = ' ';
    // se la stringa non è vuota, ci sarà almeno una riga
    counter->lines++;
    // iteriamo su tutti i caratteri della stringa
    while (*str != '\0')
        // a ogni carattere incrementiamo il contatore dei caratteri
        counter->chars++;
        if (*str == '\n')
            // se il carattere è un carattere di nuova riga incrementiamo
            // il contatore delle righe
            counter->lines++;
        }
        if (is_empty(*str) && !is_empty(prev))
            // se il carattere corrente è uno spazio e il carattere precedente
            // non era uno spazio, incrementiamo il contatore delle parole
            counter->words++;
        }
        // salviamo il carattere corrente per il prossimo ciclo
        prev = *str;
        str++;
    }
    // se l'ultimo carattere della stringa non è uno spazio
    // non è stato incrementato il contatore delle parole
    // quindi è necessario effettuare un controllo aggiuntivo
    // per incrementare il contatore delle parole
    if (!is_empty(prev))
        counter->words++;
    }
```