```
//
//  Ansi.c
//  All
//
//  Created by Filippo Fontanelli on 02/04/11.
//  Copyright 2011 __MyCompanyName__. All rights reserved.
//

#include "Ansi.h"
static char* hypertext;

int SetAnsi(token * t,queue *q,char *app){
    int i,emoticon;
/*creo la parte iniziale della formattazione*/
    strcat(app, ESCapeStart);
    strcat(app, RIPRISTINA);
    strcat(app, ESCapeF);

    i = 0;
    emoticon = 0;
    /*GRASSETTO*/
    if (t->formattazione[FS_B][FORMATTAZIONE] == 1){
        strcat(app, GRASSETTO);
        strcat(app, ESCapeF);
        i++;
    }
    /*CORSIVO*/
    if (t->formattazione[FS_I][FORMATTAZIONE] == 1){
        strcat(app, CORSIVO);
        strcat(app, ESCapeF);
        i++;
    }
    /*SOTTOLINEATO*/
    if (t->formattazione[FS_U][FORMATTAZIONE] == 1){
        strcat(app, SOTTOLINEATO);
        strcat(app, ESCapeF);
        i++;
    }
    /*CITAZIONE*/
    if (t->formattazione[FE_QUOTE][FORMATTAZIONE] == 1){
        strcat(app, GIALLO);
        strcat(app, ESCapeF);
        i++;
    }
    /*FINE PARAGRAFO*/
    if (t->formattazione[F_P][FORMATTAZIONE] == 1){
        i++;
        t->formattazione[F_P][FORMATTAZIONE] = 0;
    }
    /*IMMAGINE*/
    if (t->formattazione[FS_IMG][FORMATTAZIONE] == 1){
        strcat(app, SOTTOLINEATO);
        strcat(app, ESCapeF);
        i++;
    }
    /*HYPERLINK*/
    if (t->formattazione[FS_URL][FORMATTAZIONE] == 1){
        strcat(app, CORSIVO);
        strcat(app, ESCapeF);
        i++;
    }
    /*SAD*/
    if (t->formattazione[F_SAD][FORMATTAZIONE] == 1){
        emoticon++;
```



```
        t->formattazione[F_SAD][FORMATTAZIONE] = 0;
    }
    /*SMILE*/
    if (t->formattazione[F_SMILE][FORMATTAZIONE] == 1){
        emoticon++;
        t->formattazione[F_SMILE][FORMATTAZIONE] = 0;
    }

    /*chiudo la formattazione*/
    if (app[strlen(app)-1] == ';')
        app[strlen(app)-1] = 'm';
    else
        strcat(app, ESCapeEnd);

    return ((i > 0 || emoticon != 0));
}


int PrintAnsi(queue *q, int *endline){
    token *t;
    char *stampa;
    char *out;
    int i,sum,n;

    sum =0;
    t = front(q);
    n = 0;

    while (t != NULL) {
        n = 0,i =0;

        MALLOC(out, sizeof(char)*100);
        MALLOC(stampa, sizeof(char)*LINK);
        SetAnsi(t,q,stampa);
        strcat(stampa, "%s");

        /*Analizzo la stringa del token, stampando solo un numero
         *di caratteri pari alla lunghezza della colonna, in caso di ulteriori
         *caratteri saranno preceduti da un \n
         */
        while(t->stringa[n] != '\0'){
            if ( t->stringa[n] == '\n')
                *endline= 0;
            if (*endline == q->want_opt[O_COLUM]+1){
                out[i++] = '\n';
                *endline = 0;
            }
            out[i++] = t->stringa[n];
            n++;
            (*endline)++;
        }
        out[i] = '\0';
        fprintf(stdout,stampa, out);
        /*ripristino la formattazione*/
        fprintf(stdout,RIPRISTINAEND);
        sum += strlen(out);
        t = t->next;

        free(stampa);
        free(out);
    }

    return sum;
}
```

```
static int check_smile(token *t,char* p){

    if(!(strcmp(p,S_SMILE))){
        strcat(t->stringa,":-)");
        t->formattazione[F_SMILE][FORMATTAZIONE] = 1;
        return EXIT_SUCCESS;
    }
    if(!(strcmp(p,S_SAD))){
        strcat(t->stringa,":-(");
        t->formattazione[F_SAD][FORMATTAZIONE] = 1;
        return EXIT_SUCCESS;
    }

    return EXIT_FAILURE;
}

static int Check(token *t, char *pch){
    char *link;

    if (t->formattazione[FS_CODE][FORMATTAZIONE] == 1)
        if (!(strcmp(pch, E_CODE))){
            t->formattazione[FE_CODE][FORMATTAZIONE] = 0;
            t->formattazione[FS_CODE][FORMATTAZIONE] = 0;
        }
        else
            return EXIT_FAILURE;

    /*START DEFINION*/
    if (!(strcmp(pch, S_B)) && t->formattazione[FS_B][FORMATTAZIONE] == 0){
        t->formattazione[FS_B][FORMATTAZIONE] = 1;
        return EXIT_SUCCESS;
    }
    if (!(strcmp(pch, S_I)) && t->formattazione[FS_I][FORMATTAZIONE] == 0){
        t->formattazione[FS_I][FORMATTAZIONE] = 1;
        return EXIT_SUCCESS;
    }
    if (!(strcmp(pch, S_U)) && t->formattazione[FS_U][FORMATTAZIONE] == 0){
        t->formattazione[FS_U][FORMATTAZIONE] = 1;
        return EXIT_SUCCESS;
    }
    if (!(strcmp(pch, S_P))){
        strcat(t->stringa, "\n");
        return EXIT_SUCCESS;
    }
    /*if (!(strcmp(pch, S_QUOTE)) && t->formattazione[FS_QUOTE][FORMATTAZIONE]
        == 0){
     t->formattazione[FS_QUOTE][FORMATTAZIONE] = 1;
     t->formattazione[FS_QUOTE][STAMPA_F] = 0;

     return EXIT_SUCCESS;
     }

     if (!(strcmp(pch, S_CODE)) && t->formattazione[FS_CODE][FORMATTAZIONE] ==
         0){
     t->formattazione[FS_CODE][FORMATTAZIONE] = 1;
     t->formattazione[FS_CODE][STAMPA_F] = 0;

     return EXIT_SUCCESS;
     }
     */

    /*END DEFINITON*/
```



```
    if (((!(strcmp(pch, E_B))) && t->formattazione[FS_B][FORMATTAZIONE] == 1){
        t->formattazione[FS_B][FORMATTAZIONE] = 0;
        return EXIT_SUCCESS;
    }
    if (!(strcmp(pch, E_I)) && t->formattazione[FS_I][FORMATTAZIONE] == 1){
        t->formattazione[FS_I][FORMATTAZIONE] = 0;
        return EXIT_SUCCESS;
    }
    if (!(strcmp(pch, E_U)) && t->formattazione[FS_U][FORMATTAZIONE] == 1){
        t->formattazione[FS_U][FORMATTAZIONE] = 0;
        return EXIT_SUCCESS;
    }
    /*if (!(strcmp(pch, E_CODE)) && t->formattazione[FS_CODE][FORMATTAZIONE] ==
        1){
     t->formattazione[FE_CODE][FORMATTAZIONE] = 1;
     t->formattazione[FE_CODE][STAMPA_F] = 0;
     t->formattazione[FS_CODE][FORMATTAZIONE] = 0;
     return EXIT_SUCCESS;
     }

     if (!(strcmp(pch, E_QUOTE)) && t->formattazione[FE_QUOTE][FORMATTAZIONE] =
         = 0){
     t->formattazione[FE_QUOTE][FORMATTAZIONE] = 1;
     t->formattazione[FE_QUOTE][STAMPA_F] = 0;
     return EXIT_SUCCESS;
     }*/

    /*URL IMG DEFINITION*/
    if((link = strpbrk(pch,"=")) != NULL){
        memcpy(t->link, link+1, strlen(link)-1);
        MALLOC(hypertext, sizeof(char)*LINK)
        strcpy(hypertext, link+1);
        if ((strstr(pch,S_URL) != NULL) && t->formattazione[FS_URL]
            [FORMATTAZIONE] == 0){
            t->formattazione[FS_URL][FORMATTAZIONE] = 1;
        }

        if ((strstr(pch,S_IMG) != NULL) && t->formattazione[FS_IMG]
            [FORMATTAZIONE] == 0){
            t->formattazione[FS_IMG][FORMATTAZIONE] = 1;
            strcat(t->stringa, "[");
            strcat(t->stringa, t->link);
            strcat(t->stringa, "]");
        }
        return EXIT_SUCCESS;
    }
    if ((strstr(pch,E_URL) != NULL) && t->formattazione[FE_URL][FORMATTAZIONE]
        == 0){
        t->formattazione[FS_URL][FORMATTAZIONE] = 0;
        strcat(t->stringa, "(");
        strcat(t->stringa, hypertext);
        strcat(t->stringa, ")");

        return EXIT_SUCCESS;
    }
    if ((strstr(pch,E_IMG) != NULL) && t->formattazione[FE_IMG][FORMATTAZIONE]
        == 0){
        t->formattazione[FS_IMG][FORMATTAZIONE] = 0;
        return EXIT_SUCCESS;
    }

    return check_smile(t, pch);
}
```

```
int CheckAnsi(token *t, queue *q ,char *pch){
    return (Check(t, pch) == EXIT_SUCCESS);
}
```