動画再生ソフト

Nitubeの開発

簡易仕様

・動画投稿サイトの機能を模したソフトウェアを作成する

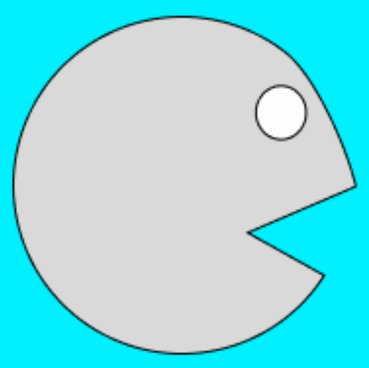
＝機能＝

◇動画データの取得と保存

◇動画タイトルの変更、並び替え

◇動画の再生

ソフトウェア概要



>動画再生中…
>> 0:停止、1:次へ、2:戻る

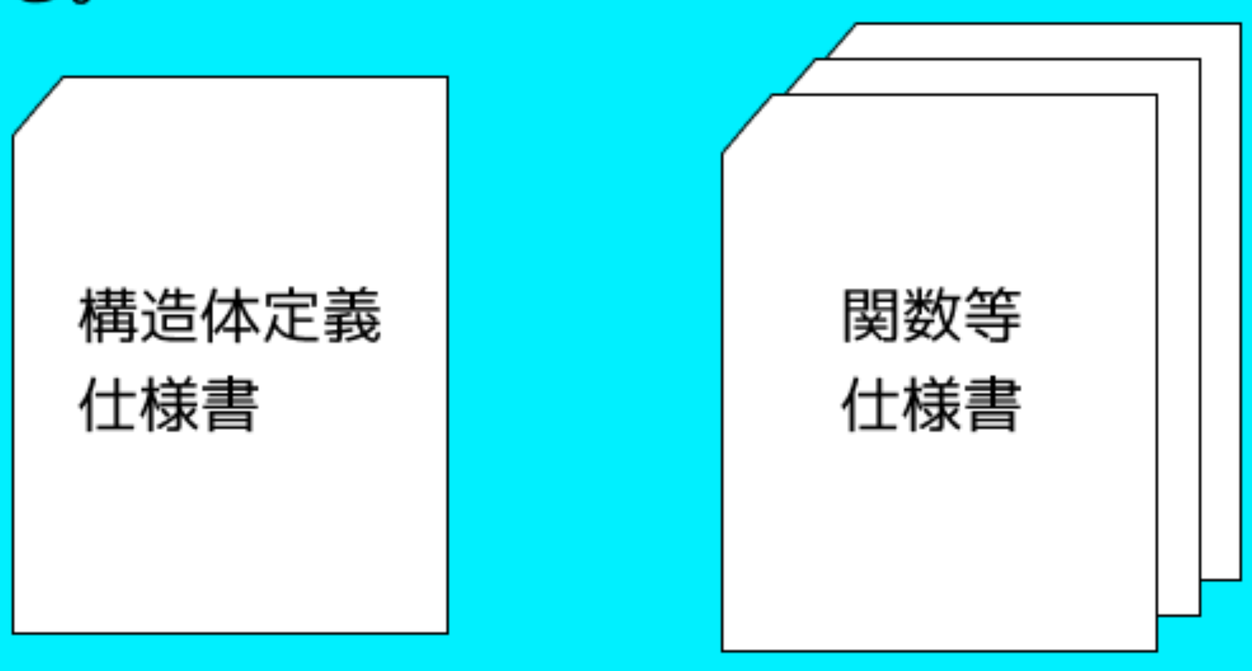
・動画投稿サイトでよく見られる機能が限定的に実装されている。

・動画自体は再生せず指定された動画情報を引き出して文字列等で動画再生中ということを表す。

簡易仕様

ソフトウェア実装概要

・動画関連情報はすべて文字列情報として構造体等メモリに格納されているとする
・ユーザインタフェースはCUIとして実装する
・各機能（ヘッダファイルに宣言されている情報）は別途関数ごとに仕様を作成している。仕様が満たされるように作成する。



簡易設計書

NITUBEメイン機能

MAIN()
NITUBE_MAIN_MENU()

MAIN()
=引数=
なし
=返り値=
0
=概要=
プログラムが実行された際に最初に呼び出される。

まずプログラム起動時にNITUBE型の構造体変数nitubeを宣言し、これを指すポインタ値を引数としてnitube_load()関数を呼び出す。
nitube_load()関数はプログラムの動作に必要となるデータを変数nitubeに戻り値として格納する。

次にnitube_main_menu()で示されるメニューを
対話的に選択し対応する関数を呼び出す。

・動画再生：nitube_play()
・動画リストの編集：nitube_edit_original()
・マイリストの編集：nitube_edit_mylist()
・データの保存：nitube_save()

各関数には引数として変数nitubeのポインタ値を渡す。

=関連関数=
呼び出し先:nitube_play(), nitube_edit_original(),
nitube_edit_mylilst(), nitube_save(), nitube_load(),
nitube_main_menu()

NITUBE_MAIN_MENU()
=引数=
なし
=返り値=
なし
=概要=
メイン関数が呼び出された際にメインメニューを表示する。
この関数をメイン関数が呼び出しメニュー一覧を
表示する。

=== メインメニュー ===
1... 動画再生
2... 動画リストの編集
3... マイリストの編集
4... データの保存
0... プログラムの終了
番号を入力してください >

表示機能のみ実装すること。

=関連関数=
呼び出し元：main()

NITUBEデータ機能

NITUBE_LOAD()
NITUBE_SAVE()

NITUBE_LOAD()
=引数=
pt_nitube...main関数で宣言した構造体変数nitubeを指すポインタ
=返り値=
なし
=概要=
プログラムの動作に必要となるデータをファイルより取得する２つの関数をよびだす。
 1. nitube_load_original()で動画リストデータを呼び出す。
引数はpt_nitubetとする。データファイルを読み取り動画リストの連結リストを構成しその先頭要素を指すポインタをpt_nitube->startに、要素数をpt_nitube->maxに格納する。

 2. nitube_load_mylist()でマイリストデータを呼び出す。
引数はpt_nitubeとする。データファイルを読み取りマイリストの連結リストを構成しその先頭要素を指すポインタをpt_nitube->mylist1.startに、要素数をpt_nitube->mylist1.maxに格納する。

=関連関数=
呼び出し元：main()
呼び出し先：nitube_load_original(), nitube_load_mylist()

NITUBE_SAVE()
=引数=
pt_nitube...main関数で宣言した構造体変数nitubeを指すポインタ
=返り値=
なし
=概要=
プログラムの動作に必要となるデータをファイルに保存するための２つの関数をよびだす。
 1. nitube_save_original()で動画リストデータを保存する。
引数はpt_nitubeとする。動画リストの連結リストをデータファイルに指定されたフォーマット、拡張子で保存する。

 2. nitube_save_mylist()でマイリストデータを保存する。
引数はpt_nitubeとする。マイリストの連結リストをデータファイルに指定されたフォーマット、拡張子で保存する。

=関連関数=
呼び出し元：main()
呼び出し先：nitube_save_original(), nitube_save_mylist()

簡易設計書

NITUBEデータ機能

NITUBE_LOAD_ORIGINAL()
NITUBE_LOAD_MYLIST()

NITUBE_LOAD_ORIGINAL()

=引数=
pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ

=返り値=
なし

=概要=
1データファイル original.dat に格納された情報を、オリジナルの動画リストとして取得することが目的．取得したデータはmalloc()関数を用いて構成したMOVIE型の連結リストに格納する．

オリジナルのデータを格納した連結リストの先頭要素を指すポインタ値をpt_nitube->startに格納し、取得した動画数をpt_nitube->maxに格納する．

MOVIE型のmylistlinkはここでは利用しないのでNULLを代入しておく

=関連関数=
呼び出し先：なし
呼び出し元：nitube_load()

NITUBE_LOAD_MYLIST()

=引数=
pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ

=返り値=
なし

=概要=
データファイル mylist1.dat に格納された情報をマイリストの動画データとして取得することが目的．取得したデータはmalloc()関数を用いて構成したMOVIE型の連結リストに格納する．

マイリスト用のdatファイルには
・マイリスト名：文字列
・格納された動画数：int値
・格納されている動画がオリジナル（動画リスト）の連結リスト上において先頭から何番目に格納されているかを表す番号：int値
のフォーマットで記録されている．

またオリジナルの連結リストとマイリストの連結リストではMOVIE型の構造体変数への動画データの格納方法が異なるので注意が必要である．具体的には動画投稿日、動画タイトル、チャンネル名、タグなどのメンバ変数には値を格納せずオリジナルでは利用しなかったmylistlinkというMOVIEポインタ型のメンバが対応するオリジナルの構造体変数を指すようにする．このように引数として受けとったピンタ変数pt_nitubeよりオリジナルの連結リストにアクセスしてデータファイルmylist1.datalより取得したint値番目の要素へのポインタ値をmylistlinkに代入する処理を行えばよい．マイリストのデータを格納した連結リストの先頭要素を指すポインタ値はpt_nitube->mylist.startに格納し読み込んだ動画の数はpt_nitube->mylist.maxに格納する．

=関連関数=
呼び出し元：nitube_load()
呼び出し先：なし

NITUBEデータ機能

NITUBE_SAVE_ORIGINAL()
NITUBE_SAVE_MYLIST()

NITUBE_SAVE_ORIGINAL()

=引数=
pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ

=返り値=
なし

=概要=
オリジナルのデータをデータファイルoriginal.datalに保存する関数

=関連関数=
呼び出し元：nitube_save()
呼び出し先：なし

NITUBE_SAVE_MYLIST()

=引数=
pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ

=返り値=
なし

=概要=
マイリストのデータをデータファイル mylist1.dat に保存する関数．次の2点に注意する．

１．マイリストの動画データはオリジナルとは異なる方法で連結リストに格納されていることに注意して保存処理を行う．
２．マイリストの動画データファイルはオリジナルと異なる形式で保存する．つまり、マイリストに格納されている動画がオリジナルの連結リスト上では何番目に格納されいているか示すint値を示す．

この値はnitube_movie_count_index関数で調べる．この関数は第１引数としてオリジナルの連結リストの戦闘要素を指すポインタ値(pt_nitube->start)、第２引数として順番を調べたいまいリストの要素がmylistlinkで指しているポインタ値を渡す事によりカウントした番号を戻り値として返す．
この値をマイリストのすべての要素について調べ順にデータファイルに保存すればよい．

=関連関数=
呼び出し元：nitube_save()
呼び出し先：nitube_movie_count_index()

簡易設計書

NITUBE再生機能

NITUBE_PLAY()
NITUBE_PLAY_MENU()

NITUBE_PLAY()

=引数=

pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ

=返り値=

なし

=概要=

再生処理の準備を行う関数．
１．まずnitube_play_menu()で表示されるメニューより動画再生の方法を対話的に選択する．
２．メニュー１を選択した場合にはnitube_play_makebuffer_mylist()を呼び出して再生バッファを用意する．引数はpt_nitube．戻り値はなし．
３．メニュー２～５を選択した場合、nitube_play_makebuffer_original()を呼び出して再生バッファを用意する．第１引数はpt_nitube、第２引数は再生方法を指定するためのint値．再生方法等の詳細はこの関数の項を参照．
４．nitube_playing()を呼び出して再生を行う．第１引数はpt_nitube、第2引数はオリジナルを再生する場合には１、マイリストを再生する場合には２を与える．

=関連関数=

呼び出し元：main()
呼び出し先：nitube_play_menu(),
nitube_play_makebuffer_original(),nitube_play_makebuffer_mylist(),nitube_playing()

NITUBE_PLAY_MENU()

=引数=

なし

=返り値=

なし

=概要=

10以下のメニューを表示する機能のみを実装する．

==== Playing menu ====
1... マイリストを再生する
2... 動画リストを再生する
3... 動画リストの指定した動画投稿日で再生する
4... 動画リストの指定したチャンネル名で再生する
5... 動画リストの指定したタグで再生する
0... メイン画面に戻る
番号を入力 >

=関連関数=

呼び出し元：nitube_play()
呼び出し先：なし

NITUBE再生機能

NITUBE_PLAY_MAKEBUFFER_ORIGINAL()
NITUBE_PLAY_MAKEBUFFER_MYLIST()

NITUBE_PLAY_MAKEBUFFER_ORIGINAL()

=引数=

pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ
rule… 再生を行う方法を指定するint値

=返り値=

なし

=概要=

オリジナルを格納した連結リストから再生バッファを準備する関数．まず引数ruleの値を参照して、
１→全再生
２→指定した投稿日のみを再生
３→指定したチャンネル名のみを再生
４→指定したタグのみを再生
値に応じて再生する準備を行う．
動画の再生には再生バッファというポインタ配列を利用する．準備として再生バッファ（MOVIEポインタ配列）に再生したいオリジナル動画（連結リストの要素）を指すポインタ値を再生順に格納する．

メニュー１を選択した場合にはオリジナルの連結リストに格納されたすべての要素を指すポインタ値を再生バッファに格納する．
メニュー２～４を選択した場合にはこの関数内においてscanf()などを用いて指定した投稿日、タグ名などの情報を取得して．my_strcmp()などをりようすることにより指定した動画のみを選んで再生バッファにぽいんた値を格納する．
最後に格納した動画数をpt_nitube->playbuffer.maxに代入する．

=関連関数=

呼び出し元：
呼び出し先：

NITUBE_PLAY_MAKEBUFFER_MYLIST()

=引数=

pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ

=返り値=

なし

=概要=

マイリストを格納した連結リストから再生バッファを準備する関数．
nitube_play_makebuffer_original()に似ているがこっちのほうが簡単．

マイリストを格納した連結リストの戦闘要素を指すポインタはpt_nitube->mylist1.startに格納されているので、これを利用してリスト要素に格納されている動画データを指すポインタ値を再生バッファに順に格納する．

格納した動画数をpt_nitube->playbuffer.maxxに代入する．

=関連関数=

呼び出し元：nitube_play()
呼び出し先：なし

簡易設計書

NITUBE再生機能

NITUBE_PLAYING()
NITUBE_PLAYING_MENU()

NITUBE_PLAYING()

=引数=

pt_nitube…main関数で宣言した構造体変数nitubeを指すポインタ
type … 再生する動画データがオリジナルかマイリストか指定する引数．
１：オリジナル、２：マイリスト　として処理する．

=返り値=

なし

=概要=

10動画の再生を表現する関数．main()関数で宣言した構造体変数nitubeにおいて再生バッファ（pt_nitube->playbuffer.list配列）の中にこれから再生する動画データを指すポインタ値が格納されている．

まずはじめにlist[0]の指すアルバムデータを再生しnitube_playing_menu()で示したメニューを対話的に選択することで次の動画、前のアルバムの再生を行う．また、再生中止を選択した場合は処理の流れをnitube_play()に返す．

存在しない動画を再生しようとした場合は（つまり、list[-1]やpt_nitube->playbuffer.maxより大きい動画を再生しようしたとき）再生を終了し処理の流れをnitube_play()に返す．

実際に動画を流すのは難しいのでここでは再生中の動画に関する情報を次のように画面に表示する．

>> Now Playing：　「再生中の動画名」 <<

=関連関数=

呼び出し元：nitube_play()
呼び出し先：nitube_playing_menu()

NITUBE_PLAYING_MENU()

=引数=

なし

=返り値=

なし

=概要=

画面に次のメッセージを表示する関数
>> 0:STOP, 1:PREV, 2:NEXT <<

=関連関数=

呼び出し元：nitube_playing()
呼び出し先：なし

簡易設計書

NITUBE共通関数機能

NITUBE_MOVIE_COUNT_INDEX()

NITUBE_MOVIE_COUNT_INDEX()

=引数=

start... MOVIE型の構造体変数を指すポインタ
count... MOVIE型の構造体変数を指すポインタ

=返り値=

countが指す要素が先頭から何番目であるかを示すint値

=概要=

countが指す要素が、startが指す連結リスト上において先頭から何番目にあるかを調べる関数.
startからポインタ値をひとつずつ次の要素に進めていき、countのポインタ値と一致するまでにループを進めた回数を数えればよい.

※注意※
カウントする要素と動画データはオリジナルとマイリストいずれの場合でも同じ処理で対応可能であること.

=関連関数=

呼び出し元：nitube_save_mylist(), nitube_edit_original_delete(), nitube_edit_swap(), nitube_edit_mylist_delete()
呼び出し先：なし

NITUBE共通関数機能

NITUBE_EDIT_SWAP()

NITUBE_EDIT_SWAP()

=引数=

start... 入れ替え操作の対象となる要素が格納された連結リストの戦闘要素を指すポインタ値

=返り値=

なし

=概要=

連結リスト内のふたつの動画を指定して、これらの順序を入れ替える関数.
１．まずnitube_movie_select()を２回実行して、ふたつの要素を指すポインタ値を取得する.
２．つぎに、nitube_movie_count_index()を利用して、１．で取得した要素が連結リスト上で先頭から何番目にあるか調べる.
この関数の戻り値として順番を意味するint値が得られるので、適当な変数に格納する.
３．得られた番号を参照して隣り合っている場合：nitube_edit_swap_otonari()、隣り合っていない場合：nitube_edit_swap_no_otonari()を呼び出す.

※注意※
・startの指す動画データはオリジナル、マイリストのどちらの場合も対応すること.
・入れ替えのキャンセル処理も実装すること

=関連関数=

呼び出し元：nitube_edit_original(), nitube_edit_mylist()
呼び出し先：nitube_edit_swap_otonari(), nitube_edit_swap_no_otonari(), nitube_movie_select(), nitube_movie_count_index()

簡易設計書

NITUBE共通関数機能

NITUBE_EDIT_SWAP_NO_OTONARI()
NITUBE_EDIT_SWAP_OTONARI()

NITUBE_EDIT_SWAP_NO_OTONARI()

=引数=

start... 入れ替え操作を行う要素が格納された連結された連結リストの戦闘要素を指すポインタ値
num1, num2 ... 入れ替え操作を行う要素が先頭から何番目にあるかを指定するint 値.

=返り値=

なし

=概要=

受け取った動画データリストポインタから格納されている動画順番を引数として受け取った２つのint値の動画を入れ替える.
※動画データリストはオリジナル、マイリストどちらも対応すること
※入れ替え対象となる動画は隣り合っていない動画を入れ替える

=関連関数=

呼び出し元：nitube_edit_swap()
呼び出し先：なし

NITUBE_EDIT_SWAP_OTONARI()

=引数=

start... 入れ替え操作を行う要素が格納された連結された連結リストの戦闘要素を指すポインタ値
num1, num2 ... 入れ替え操作を行う要素が先頭から何番目にあるかを指定するint 値.

=返り値=

なし.

=概要=

受け取った動画データリストポインタから格納されている動画順番を引数として受け取った２つのint値の動画を入れ替える.
※動画データリストはオリジナル、マイリストどちらも対応すること
※入れ替え対象となる動画は隣り合っている動画を入れ替える

=関連関数=

呼び出し元：nitube_edit_swap()
呼び出し先：なし

NITUBE共通関数機能

NITUBE_MOVIE_SELECT()
NITUBE_MOVIE_SELECT_MENU()

NITUBE_MOVIE_SELECT()

=引数=

start... 選択を行う動画が格納されている連結リストの先頭要素を指すポインタ
type... startで与えられた連結リストがオリジナルかマイリストかを区別する値．1：オリジナル、2：マイリストとして処理する.

=返り値=

MOVIE型のポインタ値... 選択したアルバムを指すポインタ値を返す．もし選択をキャンセルした場合はNULLポインタを返す.

=概要=

連結リストに格納された動画のうちから、いずれか一つの動画を選択しこれを指すポインタ値をreturnする関数．まずnitube_movie_select_menu()を用いて表示したメニュー項目を対話的に選択しその中から選択
１．．．全動画一覧を表示してその中から選択
２．．．指定した登校日の動画一覧を表示してその中から選択
３．．．指定したチャンネル名の動画一覧を表示してその中から選択
４．．．指定したタグの動画一覧を表示してその中から選択
０．．．選択することをキャンセルする.

動画一覧表示にはnitube_movie_print()を利用する．これは動画一覧の表示方法を引数によって指定することができる.

=関連関数=

呼び出し元：nitube_edit_swap(), nitube_edit_original(), nitube_edit_mylist_add(), nitube_edit_original_delete(), nitube_edit_mylist_delete()
呼び出し先：nitube_movie_select_menu(), nitube_movie_print()

NITUBE_MOVIE_SELECT_MENU()

=引数=

なし

=返り値=

なし

=概要=

動画選択のために以下のメッセージを画面に表示する関数.

=== 動画選択 ===
1... 全動画一覧より選択
2... 指定した投稿日一覧より選択
3... 指定したチャンネル名一覧より選択
4... 指定したタグ一覧より選択
0... キャンセル
番号を入力 >

=関連関数=

呼び出し元：nitube_movie_select()
呼び出し先：なし

簡易設計書

NITUBE共通関数機能

MY_SCANF_INTEGER()

=引数=
なし

=返り値=
取得したint値

=概要=
安全に整数値を取得する関数．
scanfによりいったん文字列として取得した値をatoi()関数を利用して整数型にキャストする．

※注意※
atoi()関数はstdlib.hが必要．

=関連関数=
省略．整数型のキーボード入力が必要な関数．文字列はscanf関数を利用する．

NITUBE_MOVIE_PRINT()

=引数=
start... 表示する動画が格納された連結リストを指すポインタ値
type... 表示する動画データがオリジナルとマイリストのいずれかを指定するint値．１：オリジナル、２：マイリストとして処理を区別する．
rule... どのような方法で画面に一覧を表示するかを指定するint値．
1... 全動画一覧を表示する
2... 指定した投稿日の動画一覧を表示する
3... 指定したチャンネル名の動画一覧を表示する
4... 指定したタグの動画一覧を表示する

=返り値=
なし

=概要=
動画データの一覧を画面に表示する関数．呼び出し側の関数が引数ruleの値によって表示する形式を指定する．ruleの値が2～4の場合はこの関数内でさらに必要な情報を取得する必要がある．
また引数typeの値によって表示を行う動画データがオリジナルかマイリストかを指定されるのでどちらの場合にも対応できるようにする．

=関連関数=
呼び出し元：nitube_movie_select()
呼び出し先：my_scanf_integer(), my_strcmp()

簡易設計書

ヘッダファイル例

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

#define MN 100

#define ORIGINAL "original.dat"
#define MYLIST1 "mylist.dat"

struct MOVIE
{
    int date;
    char title[MN];
    char channel[MN];
    char tag[MN];
    struct MOVIE *next, *mylistlink;

};

struct MYLIST
{
    int max;
    char name[MN];
    struct MOVIE *start;
};

struct PLAYBUFFER
{
    int max;
    struct MOVIE *list[MN];
};

struct NITUBE
{
    int max;
    struct MOVIE *start;
    struct PLAYBUFFER playbuffer;
    struct MYLIST mylist;

};
```

```
oid nitube_main_menu();
nt my_scanf_integer();
nt my_strcmp( char word1[MN], char word2[MN]);
nt strcmpare(char word1[MN], char word2[MN]);
nt nitube_movie_count_index( struct MOVIE *start, struct MOVIE *count);
oid nitube_movie_print( struct MOVIE *start, int type, int rule);
truct MOVIE *nitube_movie_select( struct MOVIE *start, int type);
oid nitube_movie_select_menu();

oid nitube_edit_original( struct NITUBE *pt_nitube);
oid nitube_edit_original_menu();
oid nitube_edit_mylist( struct NITUBE *pt_nitube);
oid nitube_edit_mylist_menu();
oid nitube_edit_original_delete( struct NITUBE *pt_nitube);
oid nitube_edit_mylist_delete( struct NITUBE *pt_nitube);
oid nitube_edit_original_add( struct NITUBE *pt_nitube);
oid nitube_edit_mylist_add( struct NITUBE *pt_nitube);
oid nitube_edit_swap( struct MOVIE *start);
oid nitube_swap_otonari( struct MOVIE *start, int num1, int num2);
oid nitube_swap_no_otonari( struct MOVIE *start, int num1, num2);

oid nitube_load( struct NITUBE *pt_nitube);
oid nitube_load_original( struct NITUBE *pt_nitube);
oid nitube_load_mylist( struct NITUBE *pt_nitube);
oid nitube_save( struct NITUBE *pt_nitube);
oid nitube_save_original( struct NITUBE *pt_nitube );
oid nitube_save_mylist( struct NITUBE *pt_nitube);

oid nitube_play( struct NITUBE *pt_nitube);
oid nitube_play_menu();
oid nitube_makebuffer_original( struct NITUBE *pt_nitube, int rule);
oid nitube_makebuffer_mylist( struct NITUBE *pt_nitube);
oid nitube_playing( struct NITUBE *pt_nitube, int type);
oid nitube_playing_menu();
```