********** BDD パッケージ（1.93 版）ドキュメント **********

著者: 湊　真一　京都大学　大学院情報学研究科

最新更新日: 2021.12.06

****

このパッケージは BDD の基本操作を行うＣ言語の関数ライブラリである。

・本プログラムは、32 ビットまたは 64 ビットの計算機で動作する。（コンパイル時に、オプション B_64 を指定すると 64 ビットモードとなる）

・各操作はＣ言語の関数呼び出しにより実行される。
関数等の宣言部は、bddc.h にあるので、include すること。
プログラム本体は bddc.c にある。

・入力変数番号(VarID)は 1 から始まる unsigned int 型（bddvar 型）の整数で識別する（0 は定数を表す）。VarID の最大値は bddvarmax で参照される。デフォルトは 65535(16 ビット)。

・各 VarID ごとに BDD での上下の順位(level)の情報を保持している。Level もまた 1 から始まる bddvar 型の整数で識別する。大きい数値ほど上位の変数を表す（BDD の根に近く、先に展開される）。VarID を何も指定せずに生成した場合は VarID と同じ値の level を持つ。

・論理演算結果の BDD は、32 ビット（または 64 ビット）の unsigned int（bddp 型）のインデックスで返される。BDD は論理関数に対して一意であり、インデックスの値も BDD に対して一意である。したがって、２つの論理演算結果が等価であるかどうかは、演算結果のインデックスの値が同じかどうかを比較することで行える。

```
bddp f, g
......
if(f == g) 一致
if(f != g) 不一致
```

ただしインデックスの大小比較（f ＞ g）は意味を持たない。

・BDD のインデックスのビット構造は以下の通りである
ビット幅 32 の場合
ABBBBBBBBBBBBBBBBBBBBBBBBBBBBBC
ビット幅 64 の場合（下位 40 ビットを使用）
OOOOOOOOOOOOOOOOOOOOOOOABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBC
A: 節点 ID か定数値かを区別するフラグ。0:節点 ID　1:定数値
B: （A=0 の場合）節点番号を表す。
　（A=1 の場合）BC を合わせて定数値を表す。
　　通常の論理関数の場合、最下位ビット C 以外は常に 0。
C: 否定枝を表すフラグ。定数値の場合は奇数偶数を表す。

・BDD の節点情報を格納する記憶領域のデータ構造を以下に示す。
ビット幅 32 の場合
Node.f0_32　ABBBBBBBBBBBBBBBBBBBBBBBBBBBBBC 0 枝の BDD インデックス
Node.f1_32　ABBBBBBBBBBBBBBBBBBBBBBBBBBBBBC 1 枝の BDD インデックス
Node.nx_32　OOBBBBBBBBBBBBBBBBBBBBBBBBBBBBB N:次の節点番号（記憶管理用）
Node.varrfc RRRRRRRRRRRRRRRRVVVVVVVVVVVVVVVV R:参照カウンタ V:変数番号

ビット幅 64 の場合
Node.f0_32　BBBBBBBBBBBBBBBBBBBBBBBBBBBBBC 0 枝の BDD インデックス
Node.f1_32　BBBBBBBBBBBBBBBBBBBBBBBBBBBBBC 1 枝の BDD インデックス
Node.nx_32　BBBBBBBBBBBBBBBBBBBBBBBBBBBBB N:次の節点番号（記憶管理用）
Node.varrfc RRRRRRRRRRRRRRRRVVVVVVVVVVVVVVVV R:参照カウンタ V:変数番号
Node.f0_h8　ABBBBBBB　0 枝の BDD インデックス（上位 8 ビット）
Node.f1_h8　ABBBBBBB　1 枝の BDD インデックス（上位 8 ビット）
Node.nx_h8　OOBBBBBB　N:次の節点番号（記憶管理用）（上位 8 ビット）

・本プログラムでは、各 BDD の参照回数を記憶するカウンタ（参照
カウンタ）を用いて記憶管理を行っている。BDD をコピーする際には、
インデックスを直接代入せず、必ず bddcopy() を使用する。また、不要
になった変数は、bddfree() にて解放することにより、記憶の再利用が
行われる。

・論理演算中に記憶あふれが発生した場合は、その演算を行う前の
状態に戻し、bddnull を返す。それ以外のエラーが発生した場合は、
エラーメッセージを出力した後、異常終了する。なお、関数の引数に

bddnull を与えた場合には、基本的に何もしないで bddnull を返す。

・本プログラムでは、組合せ集合を表す Zero-suppressed BDD (ZBDD)の
処理も行う。ZBDD と BDD の節点は内部で区別されている。ZBDD 向けの
演算の引数に BDD の節点を与えた場合（またはその逆も）エラーを検出し
異常終了する。ちなみに内部での区別の仕方であるが, BDD/ZBDD では,
0 枝側は否定枝にはならないという性質があるため, 基本的に f0_32 の
最下位ビットは 0 になっているはずである. そこで, 本来 0 であるべき
最下位ビットが 1 になっていたら ZBDD 節点であることを示している。

***** 定数マクロ *****

bddvarmax　入力変数番号の最大値（通常 65535）
bddfalse　恒偽関数を指す BDD インデックス値（0x80000000）
bddtrue　恒真関数を指す BDD インデックス値（0x80000001）
bddnull　エラーを意味する BDD インデックス値（通常 0x7FFFFFFF）
bddempty　ZBDD の空集合を表す BDD インデックス値（= bddfalse）
bddsingle　ZBDD の単位元集合を表す BDD インデックス値（= bddtrue）

***** 関数 *****

---------- [1] 初期設定・入力変数番号設定 ----------

extern int bddinit(bddp initsize, bddp limitsize)

処理系を初期化し、メモリの確保を行う。プログラムの最初に必ず実行しな
ければならない。initsize で、最初にメモリを確保する BDD 節点数を指定す
る。以後、演算中にメモリを使い切った場合は、自動的にメモリの再確保が
行われる。再確保毎に節点数は 4 倍に拡大される。拡大の上限は、limitsize
によって指定できる。使用節点数が limitsize に達したときは、メモリの再
確保はそれ以上行われず、ガベジコレクションが起動され、bddfree()により
解放された空き節点が回収される。initsize は、256 より小さい値を指定した
場合は強制的に 256 に設定される。init を下回る値を limit に指定した場合は、
強制的に limit は init と同じ値に設定される。適切な limit 値は計算機の
メモリ容量に依存する。(32 ビットマシンでは 1 節点当たり約 25 バイト、

62 ビットマシンでは約 35 バイト必要とする。) bddinit()による初期化が
正常に行われた場合には、関数の値として 0 を返し、メモリ確保に失敗した
場合 1 を返す。bddinit()を複数回実行すると、前回の内容がクリアされ、
再度初期化される。

extern bddvar bddnewvar(void)

新しい入力変数を１つ生成し、その変数番号(VarID)を返す。VarID は 1 から
始まる整数で、bddnewvar()または bddnewvaroflev()を 1 回実行するごとに
1 ずつ大きな値が返る。生成した変数の BDD 展開順位(level)は、VarID と
同じ値となる。変数の個数が最大値 bddvarmax を超えるとエラーを出力して
異常終了する。

extern bddvar bddnewvaroflev(bddvar lev)

新しい入力変数を１つ生成し、その変数番号(VarID)を返す。VarID は 1 から
始まる整数で、bddnewvar()または bddnewvaroflev を 1 回実行するごとに]
1 ずつ大きな値が返る。生成した変数の BDD 展開順位(level)は、引数 lev で
指定した値となる。実行時に順位 lev の変数がすでに存在していた場合は、
lev 以上の変数を１つずつ上にずらして（level を１ずつ増加させ)、空いた
ところに新しい変数を挿入する。引数 lev は 1 以上かつ「関数実行直前の変数の
個数＋１」以下でなければならない。そうでなければエラーを出力して異常終了
する。変数の個数が最大値 bddvarmax を超えるとエラーを出力して異常終了する。

extern bddvar bddlevofvar(bddvar v)

引数 v で指定した変数番号(VarID) の BDD 展開順位(level)を返す。引数 v は
1 以上かつ「現在の変数の個数」以下でなければならない。そうでなければ
エラーを出力して異常終了する。

extern bddvar bddvaroflev(bddvar lev)

引数 lev で指定した BDD 展開順位(level)を持つ変数番号(VarID)を返す。
引数 lev は 1 以上かつ「現在の変数の個数」以下でなければならない。
そうでなければエラーを出力して異常終了する。

extern bddvar bddvarused(void)

現在の入力変数の個数を返す。

---------- [2] 基本的な論理演算 ----------

extern bddp　bddprime(bddvar v)

変数番号 v の入力変数に関する単項関数を表す BDD を作り、それを指すインデックスを返す。すでに同じ BDD が存在していれば共有し、参照カウンタの値を１増やす。引数 v は１以上かつ「現在の変数の個数」以下でなければならない。実行中に記憶あふれを起こした場合は、bddnull を返す。不当な引数を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD ではない通常の BDD を生成する。ZBDD 向け処理には利用できない。

extern bddvar bddtop(bddp f)

f が指す BDD の最上位の節点の変数番号(VarID)を返す。返すのは BDD 展開順位の値(level)ではなくその値を持つ変数の VarID であることに注意。f の参照カウンタの値は変化しない。f が定数関数の場合は 0 を返す。引数に bddnull を与えた場合は 0 を返す。不当な引数（BDD を正しく指していない等）を与えた場合はエラーを出力し異常終了する。この演算は BDD, ZBDD 共に利用可能。

extern bddp　bddcopy(bddp f)

f が指す BDD をコピーする。すなわち、参照カウンタの値を１増やし、f そのものを返す。bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は BDD,ZBDD 共に利用可能。

extern bddp　bddnot(bddp f)

f の否定を表す BDD を作り、それを指すインデックスを返す。「否定枝」を使用しているため、節点数は増加せず、f の参照カウンタの値を１増やす

だけで、定数時間で結果を返す。bddnul を与えた場合は bddnull を返す。
不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを
出力し異常終了する。この演算は ZBDD では定義されていないため、f が
ZBDD を指していた場合はエラーを出力し異常終了する。

extern bddp　bddand(bddp f, bddp g)

f と g の論理積を表す BDD を作り、それを指すインデックスを返す。計算
結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１
増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に
bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指
していない等）を与えた場合は、エラーを出力し異常終了する。この演算は
ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを
出力し異常終了する。

extern bddp　bddor(bddp f, bddp g)

f と g の論理和を表す BDD を作り、それを指すインデックスを返す。計算
結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１
増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に
bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指
していない等）を与えた場合は、エラーを出力し異常終了する。この演算は
ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを
出力し異常終了する。

extern bddp　bddxor(bddp f, bddp g)

f と g の排他的論理和を表す BDD を作り、それを指すインデックスを返す。
計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を
１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に
bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指
していない等）を与えた場合は、エラーを出力し異常終了する。この演算は
ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを
出力し異常終了する。

extern bddp　bddnand(bddp f, bddp g)

f と g の論理積の否定を表す BDD を作り、それを指すインデックスを返す。計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを出力し異常終了する。

```
extern bddp    bddnor(bddp f, bddp g)
```

f と g の論理和の否定を表す BDD を作り、それを指すインデックスを返す。計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを出力し異常終了する。

```
extern bddp    bddxnor(bddp f, bddp g)
```

f と g の排他的論理和の否定を表す BDD を作り、それを指すインデックスを返す。計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを出力し異常終了する。

```
extern bddp    bddat0(bddp f, bddvar v)
```

f が指す BDD に対して、変数番号 v の入力変数に 0 を代入したときの BDD を作り、それを指すインデックスを返す。計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合

は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f が ZBDD を指していた場合はおかしな計算をする可能性がある。

```
extern bddp    bddat1(bddp f, bddvar v)
```

f が指す BDD に対して、変数番号 v の入力変数に 1 を代入したときの BDD を作り、それを指すインデックスを返す。計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f が ZBDD を指していた場合はおかしな計算をする可能性がある。

---------- [3] 記憶管理・表示 ----------

```
extern void    bddfree(bddp f)
```

f が指す BDD がもはや不要であることを宣言する。すなわち、参照カウンタの値を１減らす。定数関数の場合は何もしない。f が指していた BDD は、ガベジコレクションが起動されるまでは、回収されずに残っている。
引数に bddnull を与えた場合は、何もしない。この演算は BDD,ZBDD 共に利用可能。

```
extern bddp    bddused(void)
```

現在使用中の総節点数を返す。bddfree() によって解放された節点も、回収されるまでは使用中として数えるため、正確な節点数を知るには、直前に bddgc()を実行（ガベジコレクション起動）する必要がある。

```
extern int     bddgc(void)
```

強制的にガベジコレクション（不要な節点の回収）を行う。bddgc()を陽に起動しなくても、記憶が足りなくなった場合には自動的に起動される。
ガベジコレクションで空き節点が回収された場合は 0 を返し、空き節点が１個も見つからなかった場合は 1 を返す。

```
extern bddp    bddsize(bddp f)
```

f が指すの BDD の節点数を返す。参照カウンタの値は変化しない。
引数に bddnull を与えた場合は、0 を返す。不当な引数（BDD を正しく
指していない等）を与えた場合は、エラーを出力し異常終了する。
この演算は BDD,ZBDD 共に利用可能。

```
extern bddp    bddvsize(bddp *p, int lim)
```

bddp 型の配列 p[]（配列長の上限 lim）により与えられた複数の BDD の節点数を返す。
複数の BDD に共通に含まれる節点は重複して数えない。参照カウンタの値は変化しない。
配列の要素として bddnull が出現したら、その直前で配列が終了しているとする。
bddnull が出現しなければ配列長は lim までとする。配列の記憶領域はあらかじめ
確保されているものとする。不当な引数を与えた場合はエラーを出力し異常終了する。
この演算は BDD,ZBDD 共に利用可能。

```
extern void    bddexport(FILE *strm, bddp *p, int lim)
```

bddp 型の配列 p[]（配列長の上限 lim）により与えられた複数の BDD の構造を、
strm で指定するファイルに出力する。配列の要素として bddnull が出現したら、
その直前で配列が終了しているとする。bddnull が出現しなければ配列長は
lim までとする。配列の記憶領域はあらかじめ確保されているものとする。
不当な引数を与えた場合はエラーを出力し異常終了する。
この演算は BDD,ZBDD 共に利用可能。

```
extern int    bddimport(FILE *strm, bddp *p, int lim)
```

strm で指定するファイルから BDD の構造を読み込み、bddp 型の配列 p[]（配列長の上限
lim）に格納する。読み込んだ BDD 配列の要素数の１つ先の要素に bddnull を書き込む。
ただし、ファイルに書かれているデータの配列長が lim より大きいときは lim まで
しか読まない。配列の記憶領域はあらかじめ確保されているものとする。
ファイルに文法誤りが合った場合等、異常終了時は 1 を返す。正常時は 0 を返す。
この演算は BDD でのみ正しく動作する。

```
extern int    bddimportz(FILE *strm, bddp *p, int lim)
```

strm で指定するファイルから ZBDD の構造を読み込み、bddp 型の配列 p[]（配列長の上限 lim）に格納する。読み込んだ ZBDD 配列の要素数の１つ先の要素に bddnull を書き込む。
ただし、ファイルに書かれているデータの配列長が lim より大きいときは lim まで
しか読まない。配列の記憶領域はあらかじめ確保されているものとする。
ファイルに文法誤りが合った場合等、異常終了時は 1 を返す。正常時は 0 を返す。
この演算は ZBDD でのみ正しく動作する。

extern void　bddgraph(bddp f)

f が指す BDD のグラフ構造を X-Window に描画する。引数に bddnull を与
えた場合は、何も表示しない。不当な引数（BDD を正しく指していな
い等）を与えた場合は、エラーを出力し異常終了する。この演算は BDD,ZBDD
共に利用可能。

extern void　bddgraph0(bddp f)

f が指す BDD のグラフ構造を X-Window に描画する。bddgraph()とほとんど
同じだが、否定枝を使わない場合のグラフ構造を描画する。引数に bddnull
を与えた場合は、何も表示しない。不当な引数（BDD を正しく指して
いない等）を与えた場合は、エラーを出力し異常終了する。
この演算は ZBDD では正しく表示されない。

extern void　bddvgraph(bddp *p, int lim)

bddp 型の配列 p[]（配列長 n）により与えられた複数の BDD ののグラフ構造
を X-Window に描画する。配列の要素として bddnull が出現したら、その直前で
配列が終了しているとする。bddnull が出現しなければ配列長は lim までとする。
配列の記憶領域はあらかじめ確保されているものとする。不当な引数
（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。
この演算は BDD,ZBDD 共に利用可能。

extern void　bddvgraph0(bddp *p, int n)

bddp 型の配列 p[]（配列長 n）により与えられた複数の BDD ののグラフ構造
を X-Window に描画する。bddvgraph()とほとんど同じだが、否定枝を使わ

ない場合のグラフ構造を描画する。配列の要素として bddnull が出現したら、
その直前で配列が終了しているとする。bddnull が出現しなければ配列長は
lim までとする。配列の記憶領域はあらかじめ確保されているものとする。
不当な引数（BDD を正しく指していない等）を与えた場合は、
エラーを出力し異常終了する。この演算は ZBDD では正しく表示されない。

---------- [4] その他の論理演算 ----------

extern bddp bddlshift(bddp f, bddvar shift)

f が指す BDD について、関係する全ての入力変数を、展開順位(level)が
shift ずつ大きい（上位にある）変数の変数番号(VarID)にそれぞれ書き換えて
BDD を複製し、それを指すインデックスを返す。実行結果において未定義の入力
変数が必要になるような shift を与えてはならない。必要な入力変数は
あらかじめ宣言しておくこと。計算結果と同じ BDD がすでに存在していれば
共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした
場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。
shift に負の値を指定することはできない。不当な引数（BDD を正しく指し
ていない等）を与えた場合は、エラーを出力し異常終了する。この演算は
BDD,ZBDD 共に利用可能。

extern bddp bddrshift(bddp f, bddvar shift)

f が指す BDD について、関係する全ての入力変数を、展開順位(level)が
shift ずつ大きい（上位にある）変数の変数番号(VarID)にそれぞれ書き換えて
BDD を複製し、それを指すインデックスを返す。実行結果において未定義の入力
変数が必要になるような shift を与えてはならない。したがって、f に関係
しない入力変数が下位レベルに用意されていなければならない。計算結果と
同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。
実行中に記憶あふれを起こした場合は、bddnull を返す。引数に bddnull を
与えた場合は、bddnull を返す。shift に負の値を指定することはできない。
不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し
異常終了する。この演算は BDD,ZBDD 共に利用可能。

extern bddp bddsupport(bddp f)

f が指す BDD に関係する入力変数（変数の値が 0 か 1 かで f の結果が異なる
ような変数）の集合を取り出す。演算結果は、関係する入力変数の論理和
(例: a + b + d) を表す BDD を生成し、それを指すインデックスを返す。
（演算結果の BDD の 0 枝を順にたどっていくと、求める変数が得られる）
f が定数の場合は bddfalse を返す。不当な引数（BDD を正しく指して
いない等）を与えた場合は、エラーを出力し異常終了する。この演算は
BDD,ZBDD 共に利用可能。f が ZBDD の場合は演算結果は ZBDD の集合和の
形式となる。

```
extern bddp   bdduniv(bddp f, bddp g)
```

全称作用演算(universal quantification)。g で指定した入力変数の部分集合
に 0,1 の定数を代入したときに、どのような 0,1 の組合せを代入しても常に f=1
となる場合には 1 を返し、それ以外は 0 を返すような論理関数の BDD を作り、
それを指すインデックスを返す。入力変数の部分集合の与え方は、bddsupport()
の場合と同様で、変数の論理和の形式とする。(g が指す BDD の 0 枝を順に
たどっていくと、求める変数が得られる。）計算結果と同じ BDD がすでに存在
していれば共有し、参照カウンタの値を 1 増やす。実行中に記憶あふれを起こした
場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。
不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し
異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を
指していた場合はエラーを出力し異常終了する。

```
extern bddp   bddexist(bddp f, bddp g)
```

存在作用演算(existential quantification)。g で指定した入力変数の部分集合
に 0,1 の定数を代入したときに、どのような 0,1 の組合せを代入しても常に f=0
となる場合には 0 を返し、それ以外は 1 を返すような論理関数の BDD を作り、
それを指すインデックスを返す。入力変数の部分集合の与え方は、bddsupport()
の場合と同様で、変数の論理和の形式とする。(g が指す BDD の 0 枝を順に
たどっていくと、求める変数が得られる。）計算結果と同じ BDD がすでに存在
していれば共有し、参照カウンタの値を 1 増やす。実行中に記憶あふれを起こした
場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。
不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し
異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を

指していた場合はエラーを出力し異常終了する。

extern bddp　bddcofactor(bddp f, bddp g)

g = 0 のときを don't care として f を簡単化した BDD を作り、それを指すインデックスを返す。計算結果と同じ BDD がすでに存在していれば共有し、参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、bddnul を返す。引数に bddnull を与えた場合は、bddnull を返す。不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを出力し異常終了する。

extern int　bddimply(bddp f, bddp g)

f -> g （f が真ならば g は真）が恒に成り立つかを調べ、恒に成り立つなら 1 を返し、1 つでも反例があれば 0 を返す。実行中に節点数は増加しないので効率が良い。
引数に bddnull を与えた場合は、0 を返す。
不当な引数（BDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。この演算は ZBDD では定義されていないため、f,g が ZBDD を指していた場合はエラーを出力し異常終了する。

extern void　bddwcache(unsigned char op, bddp f, bddp g, bddp h)

演算結果キャッシュへの登録を行う。op は演算の種類を表す。(f op g) = h という演算結果を登録する。f, g, h は、bddp 型のデータを与える。
引数エラーチェックは行っていないので注意。この演算は BDD,ZBDD 共に利用可能である。なお、1.00 版では、op = 0～9 は、BDD 処理系内部の演算用に、op = 10～19 は、ZBDD 関係の演算用に使用されており、20 以上の番号が未使用である。複数のアプリケーションで番号が衝突しないように注意が必要。

extern bddp　bddrcache(unsigned char op, bddp f, bddp g)

演算結果キャッシュを参照する。過去に同じ演算が登録されていれば、演算結果の BDD へのインデックスを返し、見つからなければ bddnull を返す。ただし、値を返すだけで、参照カウンタの処理は行わないため、

呼び出し側で bddcopy()を実行する必要がある。引数エラーのチェックは
行っていないので注意。

---------- [5] ZBDD 用の組合せ集合演算 ----------

extern bddp　bddoffset(bddp f, bddvar v)

f が指す ZBDD において、入力変数番号 v のアイテムを含まない組合せ要素
を集めた部分集合を表す ZBDD を作り、それを指すインデックスを返す。
計算結果と同じ ZBDD がすでに存在していれば共有し、参照カウンタの値を
１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。
引数に bddnull を与えた場合は、bddnull を返す。不当な引数（ZBDD を正
しく指していない等）を与えた場合は、エラーを出力し異常終了する。
この演算は ZBDD 専用であるため、f が通常の BDD を指していた場合は
エラーを出力し異常終了する。

extern bddp　bddonset(bddp f, bddvar v)

f が指す ZBDD において、入力変数番号 v のアイテムを含む組合せ要素
を集めた部分集合を表す ZBDD を作り、それを指すインデックスを返す。
計算結果と同じ ZBDD がすでに存在していれば共有し、参照カウンタの値を
１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。
引数に bddnull を与えた場合は、bddnull を返す。不当な引数（ZBDD を正
しく指していない等）を与えた場合は、エラーを出力し異常終了する。
この演算は ZBDD 専用であるため、f が通常の BDD を指していた場合は
エラーを出力し異常終了する。

extern bddp　bddonset0(bddp f, bddvar v)

bddonset() とほぼ同じであるが、抽出した部分集合の各要素から
入力変数番号 v のアイテムが取り除かれている。
bddchange(bddonset(f, rank), rank) と等価である。
v が f の最上位の変数番号であれば、1-枝の指す BDD をそのまま返す。
計算結果と同じ ZBDD がすでに存在していれば共有し、参照カウンタの値を
１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。

引数に bddnull を与えた場合は、bddnull を返す。不当な引数（ZBDD を正しく指していない等）を与えた場合は、エラーを出力し異常終了する。
この演算は ZBDD 専用であるため、f が通常の BDD を指していた場合は
エラーを出力し異常終了する。

```
extern bddp    bddchange(bddp f, bddvar v)
```

f が指す ZBDD に含まれる全ての組合せ要素について、入力変数番号 v の
アイテムの有無を反転させた組合せ集合を表す ZBDD を作り、それを指す
インデックスを返す。計算結果と同じ ZBDD がすでに存在していれば共有し、
参照カウンタの値を１増やす。実行中に記憶あふれを起こした場合は、
bddnull を返す。引数に bddnull を与えた場合は、bddnull を返す。
不当な引数（ZBDD を正しく指していない等）を与えた場合は、エラーを
出力し異常終了する。この演算は ZBDD 専用であるため、f が通常の BDD を
指していた場合はエラーを出力し異常終了する。

```
extern bddp    bddintersec(bddp f, bddp g)
```

f と g の積集合を表す ZBDD を作り、それを指すインデックスを返す。
計算結果と同じ ZBDD がすでに存在していれば共有し、参照カウンタの値を
１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に
bddnull を与えた場合は、bddnull を返す。不当な引数（ZBDD を正しく指
していない等）を与えた場合は、エラーを出力し異常終了する。この演算は
ZBDD 専用のため、f,g が通常の BDD を指していた場合はエラーを出力し
異常終了する。

```
extern bddp    bddunion(bddp f, bddp g)
```

f と g の和集合を表す ZBDD を作り、それを指すインデックスを返す。
計算結果と同じ ZBDD がすでに存在していれば共有し、参照カウンタの値を
１増やす。実行中に記憶あふれを起こした場合は、bddnull を返す。引数に
bddnull を与えた場合は、bddnull を返す。不当な引数（ZBDD を正しく指
していない等）を与えた場合は、エラーを出力し異常終了する。この演算は
ZBDD 専用のため、f,g が通常の BDD を指していた場合はエラーを出力し
異常終了する。

```
extern bddp   bddsubtract(bddp f, bddp g)
```

f と g の差集合（f に含まれ g に含まれていない要素）を表す ZBDD を作り、
それを指すインデックスを返す。計算結果と同じ ZBDD がすでに存在して
いれば共有し、参照カウンタの値を 1 増やす。実行中に記憶あふれを起
こした場合は、bddnull を返す。引数に bddnull を与えた場合は、bddnull
を返す。不当な引数（ZBDD を正しく指していない等）を与えた場合は、
エラーを出力し異常終了する。この演算は ZBDD 専用のため、f,g が通常の
BDD を指していた場合はエラーを出力し異常終了する。

```
extern bddp   bddcard(bddp f)
```

f が指す ZBDD に含まれる要素数を返す。引数に bddnull を与えた場合は、
0 を返す。参照カウンタの値は変化しない。不当な引数（ZBDD を正しく
指していない等）を与えた場合は、エラーを出力し異常終了する。
この演算は ZBDD 専用のため、f が通常の BDD を指していた場合は
エラーを出力し異常終了する。要素数が扱える数の最大値(bddnull)を
超える場合は、その最大値(bddnull)を出力する。

```
extern char   *bddcardmp16(bddp f, char *s)
```

f が指す ZBDD に含まれる要素数を最大 16 ワード長までの多倍長整数で
カウントする。結果は 16 進数文字列として s から始まる記憶領域に
格納する。s に 0(NULL)を与えて実行した場合は、必要なサイズの
文字列領域を確保(malloc)してから結果を格納し、その開始アドレスを
関数値として返す。文字列領域確保に失敗した場合は 0(NULL)を返し
終了する。0 以外の s を与えた場合は、s をそのまま関数値として返す。
0 以外の s を与える場合には、あらかじめ十分な記憶領域（64 ビット
PC の場合、最大 129 文字）を確保しておくこと。そうでない場合の
動作は保証されない。結果の格納場所が確保されていても計算途中に
メモリが不足し計算結果が不明となる場合は、空文字列を格納して
終了する。メモリは足りているが計算結果が表現可能な最大値を
超える場合は、表現可能な最大値をカウント結果として格納して終了する。
引数 f に bddnull を与えた場合は 0 をカウント結果とする。不当な引数
（ZBDD を正しく指していない等）を与えた場合は異常終了する。この演算は
ZBDD 専用のため、f が通常の BDD を指していた場合は異常終了する。

```
extern bddp   bddlit(bddp f)
```

f が指す ZBDD に含まれる組合せに出現するアイテム数の総和を返す。
引数に bddnull を与えた場合は、0 を返す。参照カウンタの値は
変化しない。不当な引数（ZBDD を正しく指していない等）を与えた
場合は、エラーを出力し異常終了する。この演算は ZBDD 専用のため、
f が通常の BDD を指していた場合はエラーを出力し異常終了する。
要素数が扱える数の最大値(bddnull)を超える場合は、その最大値
(bddnull)を出力する。

```
extern bddp   bddlen(bddp f)
```

f が指す ZBDD に含まれる組合せのうち、最もアイテム数を多く含む組合せを
探し出して、そのアイテム数を返す。引数に bddnull を与えた場合は、0 を返す。
参照カウンタの値は変化しない。不当な引数（ZBDD を正しく指していない等）を
与えた場合は、エラーを出力し異常終了する。この演算は ZBDD 専用のため、
f が通常の BDD を指していた場合はエラーを出力し異常終了する。

```
extern int bddisbdd(bddp f)
```

f が指すノードが BDD であるかを判定する。BDD なら 1 を、ZBDD なら 0 を返す。
f が定数関数の場合は 1 を返す。引数に bddnull を与えた場合は 0 を返す。
不当な引数（BDD を正しく指していない等）を与えた場合はエラーを
出力し異常終了する。この演算は BDD,ZBDD 共に利用可能。

```
extern int bddiszbdd(bddp f)
```

f が指すノードが ZBDD であるかを判定する。ZBDD なら 1 を、BDD なら 0 を返す。
f が定数関数の場合は 1 を返す。引数に bddnull を与えた場合は 0 を返す。
不当な引数（BDD を正しく指していない等）を与えた場合はエラーを
出力し異常終了する。この演算は BDD,ZBDD 共に利用可能。