TSNET

TSC

ISSN 1884-2798

目次

!

.

```
##  GA.awk (フレームワーク)  written by Ｙさ

function rnd(N){ return int(N * rand()); }  ##  乱数

# 主要変数
# P      # 個体数
# Cr     # 交叉率(%)
# Mr     # 突然変異率(%)
# G      # 世代
# DNA[]  # 遺伝子/染色体/個体
# Eval[] # 適応度
# Idx[]  # 順位

# ↓以下の各 function を本体に実装してください。
# rz_Initialization="Initialization"; # 初期集団の生成
# rz_Result="Result";                 #   結果の表示
# rz_Fitness="Fitness";               # 適応度の算出
# rz_Termination="Termination";       # 終了判定
# rz_Selection="Selection";           # 選択
# rz_Crossover="Crossover";           #   交叉
# rz_Mutation="Mutation";             # 突然変異
# rz_Regeneration="Regeneration";     # 再生 (※次世代を現世代の配列にコピー)

BEGIN{
  # ↓以下は本スクリプトにありますが、必要に応じて置き換えてください。
  rz_Evaluation="Evaluation";         # 集団の評価
  rz_Sort="sort";                     #   順位付け
  rz_Reproduction="Reproduction";     # 繁殖 (次世代の集団の生成)

  srand();
}

# ↓以下は必要に応じて、変数 rz_Setup, rz_Finish に設定して実装してください。
# function setup(){}   # 事前準備
# function finish(){}  # 後片付け



#
# メインループ
```

```
# ↓以下を本体から呼びます
function start(){
  if(rz_Setup!="") @rz_Setup();

  @rz_Initialization();
  for(G=1; @rz_Evaluation()==0; ++G)
    @rz_Reproduction();

  exit;
}
END{
  @rz_Result();
  if(rz_Finish!="") @rz_Finish();
}



#
# 集団の評価
#
function Evaluation(  t){
  for(t=1; t<=P; ++t) Eval[t]=@rz_Fitness(DNA[t]);
  @rz_Sort();
  if(G==1) @rz_Result();
  return @rz_Termination();
}
# 順位付け
function sort(  n){
  for(n=1; n<=P; ++n) Idx[n]=n;
  asort(Idx, Idx, "cmp");
}
function cmp(i1,v1,i2,v2){
  if(Eval[v1]>Eval[v2]) return -1;
  if(Eval[v1]<Eval[v2]) return 1;
  return 0;
}



#
# 繁殖（次世代の集団の生成）
#
function Reproduction(  t,nxDNA){
  delete nxDNA;
```

```
  @rz_Selection(nxDNA);
  for(t=1; t<=P; ++t){
    @rz_Mutation(t,nxDNA);
    @rz_Regeneration(DNA,nxDNA,t);
  }
}
```

```
@include "GA.awk"


##  palindromic.awk (回文数サンプル)  written by Ｙさ

BEGIN{
  rz_Setup="setup";
  rz_Finish="finish";

  rz_Initialization="Initialization"; # 初期集団の生成
  rz_Result="Result";                 #   結果の表示
  rz_Fitness="Fitness";               # 適応度の算出
  rz_Termination="Termination";       # 終了判定
  rz_Selection="Selection";           # 選択
  rz_Crossover="Crossover";           #   交叉
  rz_Mutation="Mutation";             # 突然変異
  rz_Regeneration="Regeneration";     # 再生 (※次世代を現世代の配列にコピー)

  start();
}

# 事前準備
function setup(){
  P=20;    # 個体数
  Cr=100;  # 交叉率(%)
```

```
  Mr= 10;  # 突然変異率(%)
}

# 後片付け
function finish(){
  # (おまけ)
  for(t=999; t>=100 && !isPalindromic(t*5); --t)
    ;
  printf("¥n< %d *5 = %d >¥n", t, t*5);
}



#
# 初期集団の生成
#
function Initialization(  t){
  for(t=1; t<=P; ++t) DNA[t]=rnd(900)+100;
}

# 結果の表示
function Result(  t){
  printf("¥n[%d]¥n", G);
  for(t=1; t<=P; ++t)
    printf("%2d: %d *5 = %4d (%d)¥n",
      t, DNA[Idx[t]], DNA[Idx[t]]*5, Eval[Idx[t]]);
}

#
# 適応度の算出
#
function Fitness(x){
  if(isPalindromic(x*5)) return 1000+x;
  return 0;
}
function isPalindromic(x,  n,s,p){
  s=sprintf("%d", x);
  n=length(s);
  for(p=1; p<n+1-p; ++p)
    if(substr(s,p,1)!=substr(s,n+1-p,1)) return 0;
  return 1;
}
```

```
#
# 終了判定
#
function Termination(){ return (G>=100); }


# 再生 (※次世代を現世代の配列にコピー)
function Regeneration(dst,src,t){
  dst[t]=src[t];
}



#
# 選択
#
function Selection(nxDNA,  t,n,n1,n2){
  for(t=1; t<=int(P*0.2); ++t){
    if(Eval[Idx[t]]>0) nxDNA[++n]=DNA[Idx[t]];
  }
  if(n>0 && ((P-n)%2)) nxDNA[++n]=nxDNA[1];
  for(t=n+1; t<P; t+=2){
    n1=rnd(P)+1;
    do{ n2=rnd(P)+1; }while(n1==n2);
    @rz_Crossover(n1, n2, DNA, t, t+1, nxDNA);
  }
}

#
# 交叉
#
function Crossover(p1,p2,src,c1,c2,dst,  r){
  if(rnd(100)>=Cr) return;

  r=rnd(3);
  # a. 2桁目以降 [-**]
  if(r==0){
    dst[c1]=int(src[p1]/100)*100 + src[p2]%100;
    dst[c2]=int(src[p2]/100)*100 + src[p1]%100;
  }
  # b. 3桁目のみ [--*]
  if(r==1){
    dst[c1]=int(src[p1]/10)*10 + src[p2]%10;
```

```
    dst[c2]=int(src[p2]/10)*10 + src[p1]%10;
  }
  # c. 2桁目のみ [-*-]
  if(r==2){
    dst[c1]=int(src[p1]/100)*100 + (int(src[p2]/10)%10)*10 + src[p1]%10;
    dst[c2]=int(src[p2]/100)*100 + (int(src[p1]/10)%10)*10 + src[p2]%10;
  }
}

#
# 突然変異
#
function Mutation(t,src,  r,x){
  if(rnd(100)>=Mr) return;

  r=rnd(4);  x=src[t];
  # 1桁目が変化
  if(r==0){ src[t]=(rnd(9)+1)*100 + x%100; }
  # 2桁目が変化
  if(r==1){ src[t]=int(x/100)*100 + rnd(10)*10 + x%10; }
  # 3桁目が変化
  if(r==2){ src[t]=int(x/10)*10 + rnd(10); }
  # 全桁変化
  if(r==3){ src[t]=rnd(900)+100; }
}
```

```
@include "GA.awk"


##  alphametic.awk (覆面算サンプル)  written by Ｙさ

BEGIN{
  rz_Setup="setup";
  # rz_Finish="finish";

  rz_Initialization="Initialization"; # 初期集団の生成
  rz_Result="Result";                 #   結果の表示
  rz_Fitness="Fitness";               # 適応度の算出
  rz_Termination="Termination";       # 終了判定
  rz_Selection="Selection";           # 選択
  rz_Crossover="Crossover";           #   交叉
  rz_Mutation="Mutation";             # 突然変異
  rz_Regeneration="Regeneration";     # 再生 (※次世代を現世代の配列にコピー)

  start();
}

# 事前準備
function setup(){
  P=30;   # 個体数
  Cr=80;  # 交叉率(%)
  Mr=10;  # 突然変異率(%)

  # 問題の分解
  #   WORD[], WordCnt, Top[], TopCnt,  Use[], Cnv[], GeneSize
  if(FORMULA=="") FORMULA="SEND + MORE = MONEY";
  WordCnt=split(FORMULA, WORD, /[ ¥+=]*/);

  delete ch;
  for(n=1; n<=WordCnt; ++n){
    sz=length(WORD[n]);
    for(p=1; p<=sz; ++p) ++ch[substr(WORD[n],p,1)];
  }
  delete Use;
  delete Cnv;
  for(t in ch) Use[GeneSize++]=t;           # Use[遺伝子の位置] = 使用文字
  if(GeneSize>10){
    print "impossible condition  [", GeneSize, "]";
```

```
    exit;
  }
  for(n=0; n<GeneSize; ++n) Cnv[Use[n]]=n;  # Cnv[使用文字] = 遺伝子の位置
  delete ch;
  for(n=1; n<=WordCnt; ++n) ++ch[substr(WORD[n],1,1)];
  delete Top;
  for(t in ch) Top[TopCnt++]=t;
}

# 後片付け
function finish(){
  # (おまけ)
  delete SAVE;
  for(n=0; n<10; ++n) num[n]=n;
  sequence(num,9);
}
function sequence(num,n,  p,x){
  if(n<0){
    if(wrongZero(num)==0 && calcTerm(num)==calcAns(num)){
      if(NotYetDisplay(num)){
        print "";
        for(x=0; x<GeneSize; ++x)
          printf("%s:%d ", Use[x], num[x]);
        display(num);
      }
    }
  }else{
    for(p=n; p>=0; --p){
      swap(num, n, p);
      sequence(num, n-1);
      swap(num, n, p);
    }
  }
}
function NotYetDisplay(num,  key,n){
  for(n=0; n<GeneSize; ++n) key=key num[n] "";
  if(! (key in SAVE)) return (SAVE[key]=1);
  return 0;
}


#
```

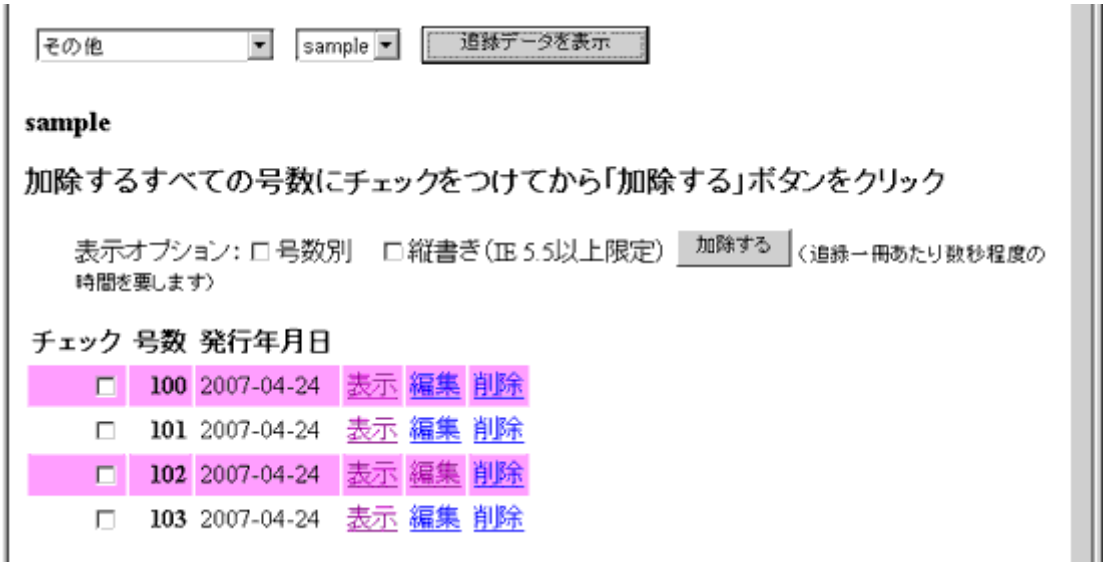
```
# 初期集団の生成
#
function Initialization(  t,n,p,x){
  for(n=0; n<10; ++n) DNA[1][n]=n;
  x=1;
  for(t=2; t<=10; ++t){
    for(n=0; n<10; ++n){
      if((p=n+x)>=10) p-=10;
      DNA[t][n]=DNA[1][p];
    }
    ++x;
  }
  random(t,DNA);
}
function random(t,tgt,  n){
  for(; t<=P; ++t){
    for(n=0; n<10; ++n) tgt[t][n]=tgt[1][n];
    for(n=0; n<10; ++n) swap(tgt[t],n,rnd(10));
  }
}
function swap(a,p1,p2,  tmp){ tmp=a[p1]; a[p1]=a[p2]; a[p2]=tmp; }

# 結果の表示
function Result(  t,n,x){
  if(GeneSize>10) exit;

  printf("¥n[%d]¥n", G);
  for(t=1; t<=P; ++t){
    printf("%2d: ", t);
    for(n=0; n<GeneSize; ++n)
      printf("%s:%d ", Use[n], DNA[Idx[t]][n]);
    printf("(%d)", Eval[Idx[t]]);
    display(DNA[Idx[t]]);
  }
}
function display(tgt,  n){
  printf("  %d", calc(tgt, WORD[1]));
  for(n=2; n<WordCnt; ++n) printf("+%d", calc(tgt, WORD[n]));
  printf("[=%d]", calcTerm(tgt));
  printf("=%d¥n", calcAns(tgt));
}
```

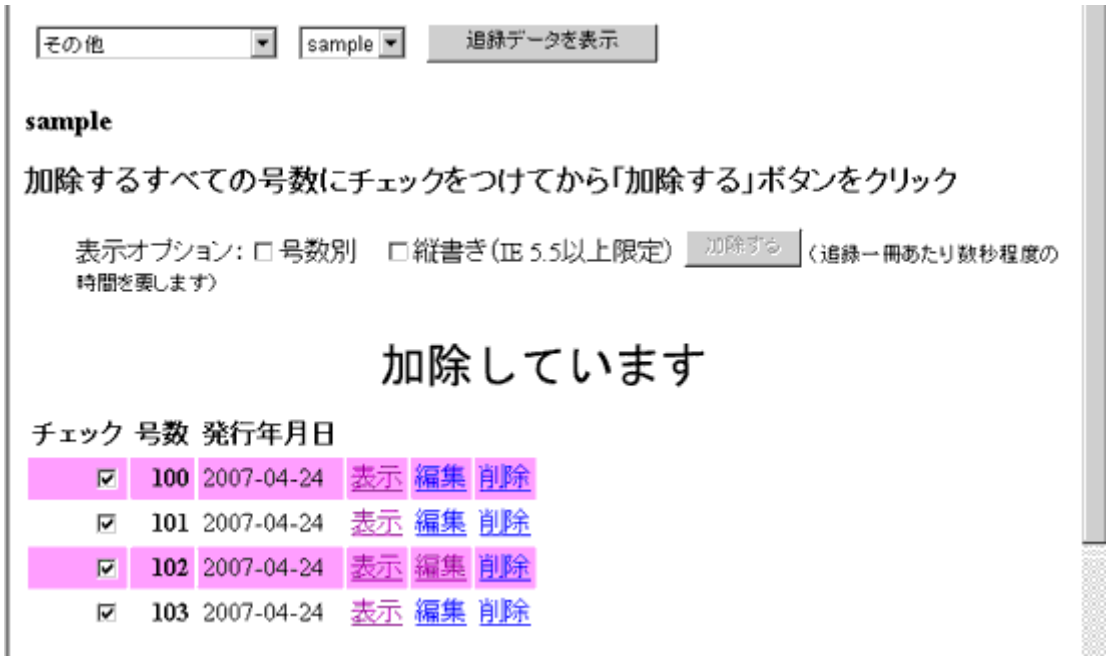
```
#
# 適応度の算出
#
function Fitness(tgt,  ev,term,ans,t,n,st,sa){
  term=calcTerm(tgt);
  ans=calcAns(tgt);
  t=wrongZero(tgt);
  ev=(TopCnt-t)*100;
  if(t==0 && term==ans) ev+=1000;
  st=term "";
  sa=ans "";
  if(length(st)==length(sa)){
    dif=0;
    for(n=1; n<=length(st); ++n){
      if(substr(st,n,1)!=substr(sa,n,1)) ++dif;
    }
    ev += int((length(st)-dif)/length(st)*100);
  }else{
    dif=(term>ans)?(ans/term):(term/ans);
    ev += int(dif*10);
  }
  return ev;
}
function calcTerm(tgt,  n,term){
  for(n=1; n<WordCnt; ++n) term+=calc(tgt,WORD[n]);
  return term;
}
function calcAns(tgt){
  return calc(tgt,WORD[WordCnt]);
}
function calc(tgt,src,  sz){
  if((sz=length(src))==0) return 0;
  return (calc(tgt,substr(src,1,sz-1))*10 + tgt[Cnv[substr(src,sz)]]);
}
function wrongZero(tgt,  n,t){
  for(n=0; n<TopCnt; ++n) if(tgt[Cnv[Top[n]]]==0) ++t;
  return t;
}
function abs(n){ return ((n>=0)?(n):(-n)); }

#
# 終了判定
```

```
#
function Termination(){
  if(MAX<Eval[Idx[1]]){
    if(Eval[Idx[1]]>1000) printf("*"); else printf(" ");
    if(prevMAX<MAX) prevMAX=MAX;
    MAX=Eval[Idx[1]];
    printf("%-11s", sprintf("[%d](%d)", G, MAX));
    display(DNA[Idx[1]]);
  }
  return (G>=300);
}


# 再生 (※次世代を現世代の配列にコピー)
function Regeneration(dst,src,t,  n){
  for(n=0; n<10; ++n) dst[t][n]=src[t][n];
}


#
# 選択
#
function Selection(nxDNA,  t,x,n,n1,n2){
  for(t=1; t<=int(P*0.2); ++t){
    ++x;
    for(n=0; n<10; ++n) nxDNA[x][n]=DNA[Idx[t]][n];
  }
  if(x>0 && ((P-x)%2)){
    ++x;
    for(n=0; n<10; ++n) nxDNA[x][n]=nxDNA[1][n];
  }
  for(t=x+1; t<int(P*0.8); t+=2){
    n1=Idx[rnd(int(P*0.3))+1];
    do{ n2=Idx[rnd(P)+1]; }while(n1==n2);
    @rz_Crossover(n1, n2, DNA, t, t+1, nxDNA);
  }
  random(t,nxDNA);
}

#
# 交叉
#
```

```
function Crossover(p1,p2,src,c1,c2,dst,  cut){
  if(rnd(100)>=Cr){
    for(n=0; n<10; ++n){
      dst[c1][n]=src[p1][n];
      dst[c2][n]=src[p2][n];
    }
    return;
  }

  cut=rnd(GeneSize-1)+1;
  if(rnd(100)>=50) cross(p1,p2,src,c1,c2,dst,   0,cut, cut);
  else             cross(p1,p2,src,c1,c2,dst, cut, 10,   0);
}
function cross(p1,p2,src,c1,c2,dst, st,ed, st2,  n,list1,list2,p,x){
  delete list1;
  delete list2;
  for(n=st; n<ed; ++n){
    list1[(dst[c1][n]=src[p1][n])]=1;
    list2[(dst[c2][n]=src[p2][n])]=1;
  }
  n=st2;
  for(p=0; p<10; ++p){ x=src[p2][p];  if(!(x in list1)) dst[c1][n++]=x; }
  n=st2;
  for(p=0; p<10; ++p){ x=src[p1][p];  if(!(x in list2)) dst[c2][n++]=x; }
}

#
# 突然変異
#
function Mutation(t,src,  n,p,x){
  if(t<3) return;

  for(n=0; n<10; ++n)
    if(rnd(100)<Mr) swap(src[t],n,rnd(10));
  return;
}
```


アドレス(D) http://localhost:3000/item/list/
移動
の加除
新しい追録名 | 追録名のリスト | 追録データ登録
分野を選択してください
追録データを表示
powered by Ruby on Rails
RAILS


その他
sample
追録データを表示
sample
加除するすべての号数にチェックをつけてから「加除する」ボタンをクリック
表示オプション: 号数別 縦書き(IE 5.5以上限定) 加除する (追録一冊あたり数秒程度の時間を要します)
チェック 号数 発行年月日
100 2007-04-24 表示 編集 削除
101 2007-04-24 表示 編集 削除
102 2007-04-24 表示 編集 削除
103 2007-04-24 表示 編集 削除

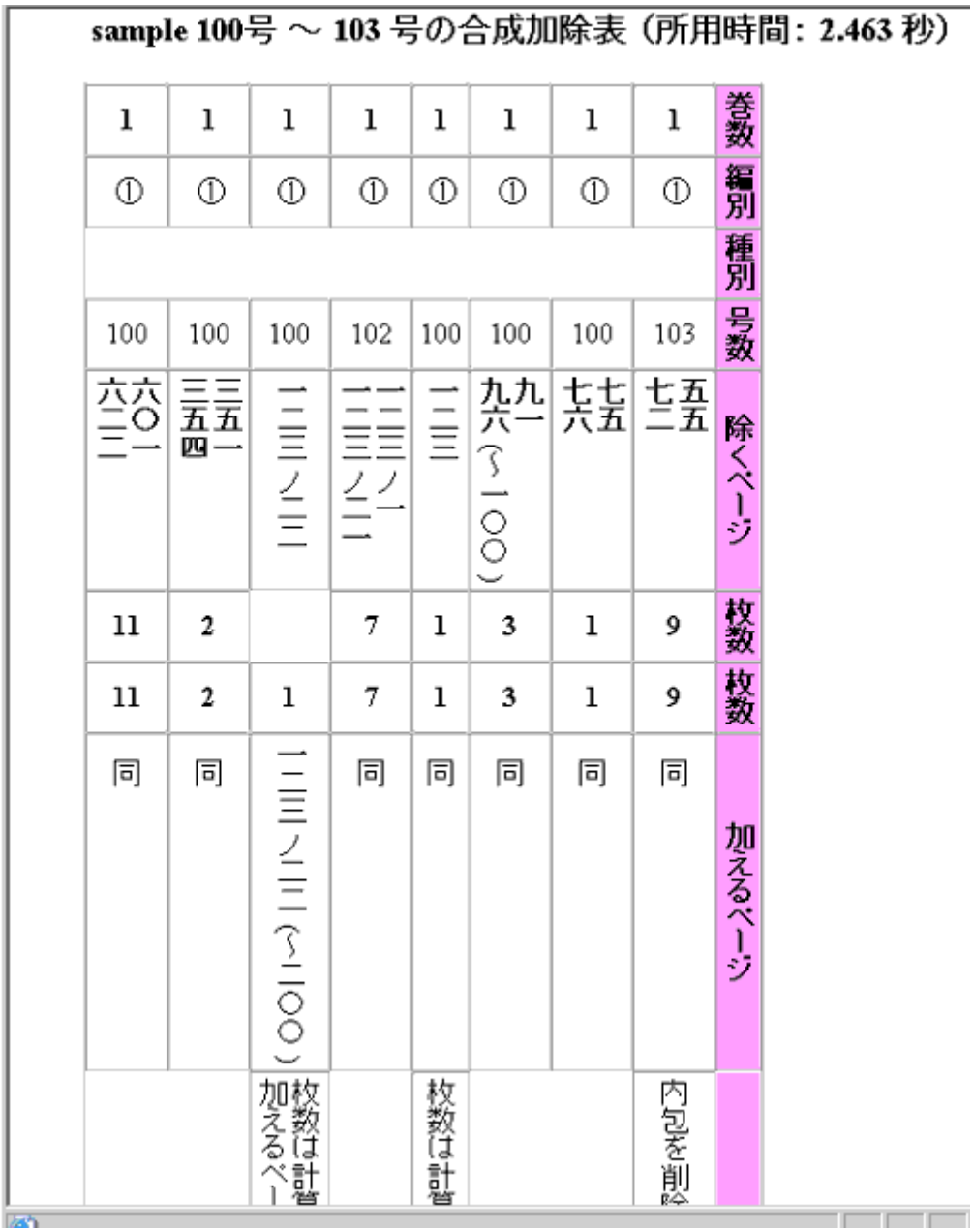


sample追録 第 100 号 加除参考表 (この追録を削除)

巻数	編別	種別	除くページ	除く枚数	加える枚数	加えるページ		
1	①		55 72	9	9	同	編集	削除
1	①		75 76	1	1	同	編集	削除
1	①		91 96 (～100)	3	3	同	編集	削除
1	①		123 123 の12 (～200)	6	11	123 123 の22 (～200)	編集	削除
1	①		351 354	2	2	同	編集	削除
1	①		601 622	11	11	同	編集	削除
除く枚数: 32 枚								

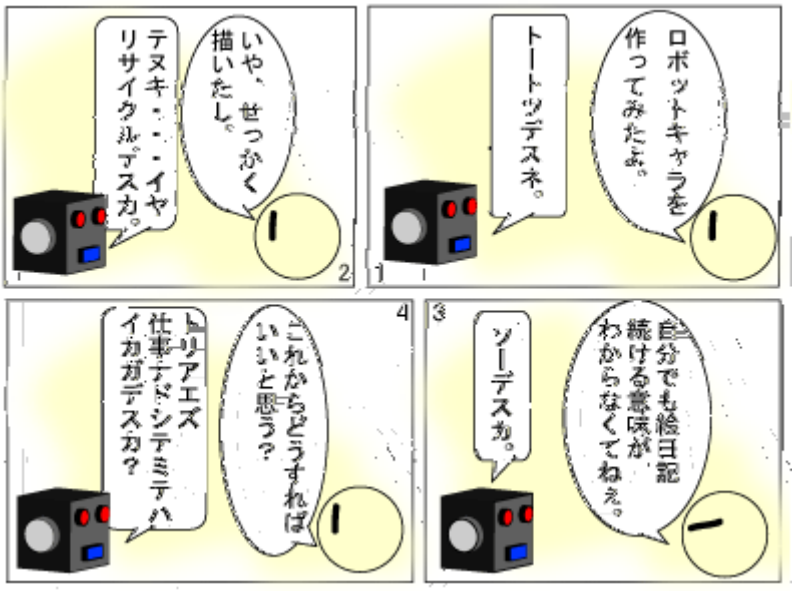
sample 100号 ～ 103 号の合成加除表（所用時間: 2.443 秒）

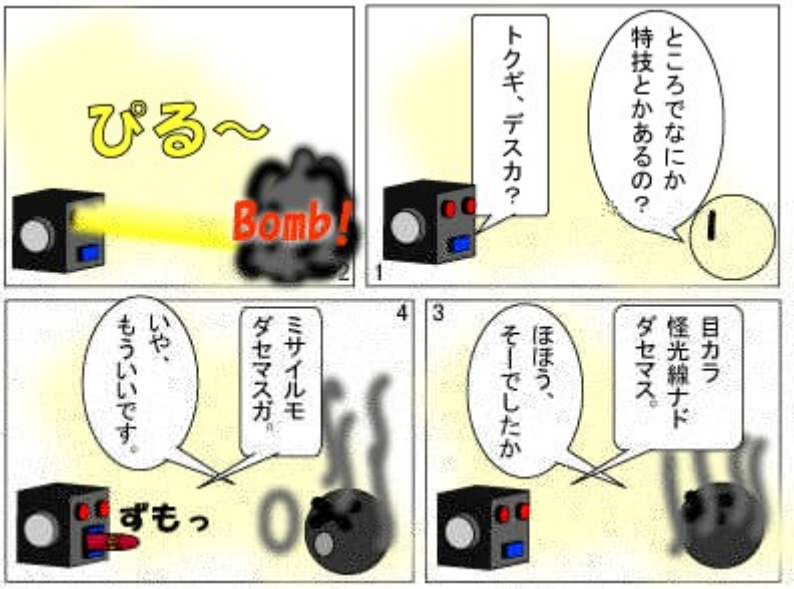
巻数	編別	種別	号数	除くページ	枚数	枚数	加えるページ	
1	①		103	55 72	9	9	同	内包を削除
1	①		100	75 76	1	1	同	
1	①		100	91 96 (～100)	3	3	同	
1	①		100	123	1	1	同	枚数は計算上
1	①		102	123 の1 123 の21	7	7	同	
1	①		100	123 の22		1	123 の22 (～200)	枚数は計算上 加えるページのみ
1	①		100	351 354	2	2	同	
1	①		100	601 622	11	11	同	
除く枚数: 34 枚 行数: 8 行 (元の除く枚数合計: 59 枚 元の行数合計: 11 行)								

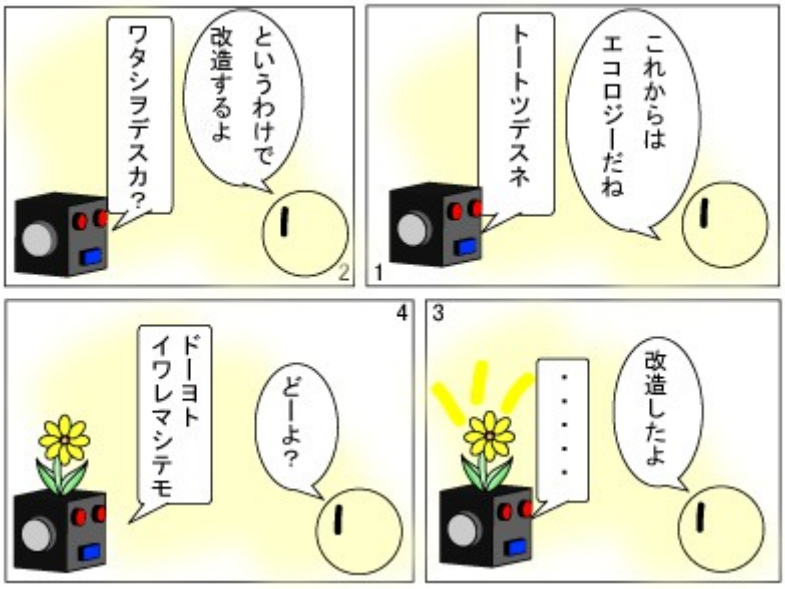


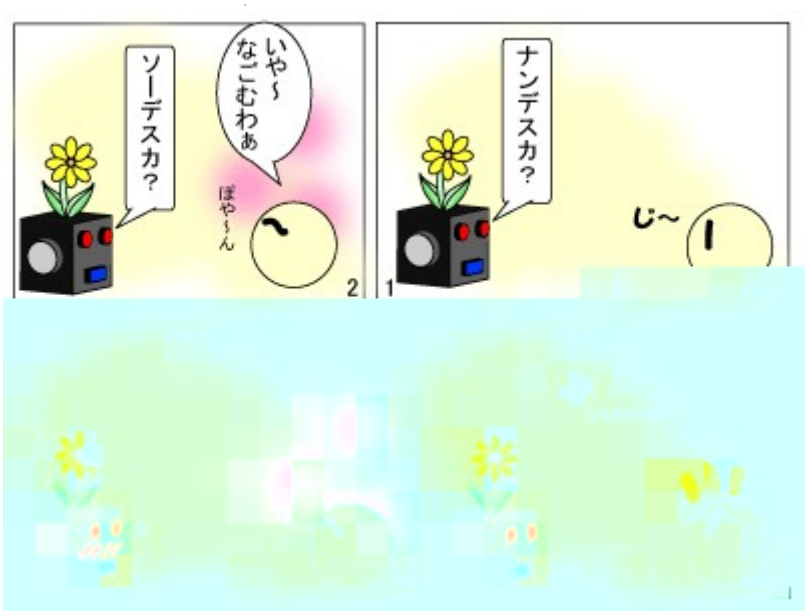
sample 100号 ～ 103 号の合成加除表（所用時間: 2.463 秒）

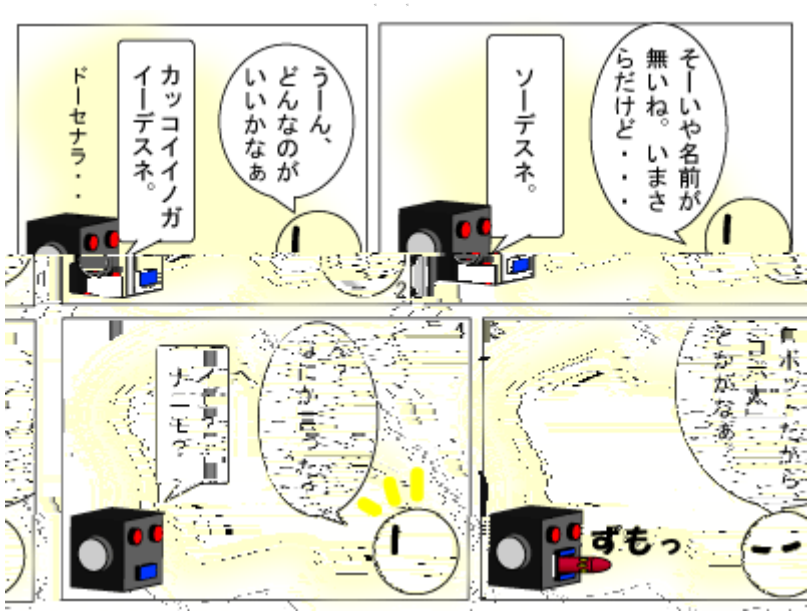
巻数	1	1	1	1	1	1	1	1
編別	①	①	①	①	①	①	①	①
種別								
号数	100	100	100	102	100	100	100	103
除くページ	六〇一 六二二	三五一 三五四	一二三ノ二三	一二三ノ一 一二三ノ二二	一二三	九一 九六（～一〇〇）	七五 七六	五五 七二
枚数	11	2		7	1	3	1	9
枚数	11	2	1	7	1	3	1	9
加えるページ	同	同	一二三ノ二三（～一〇〇）	同	同	同	同	同
			枚数は計算加えるべー		枚数は計算			内包を削除

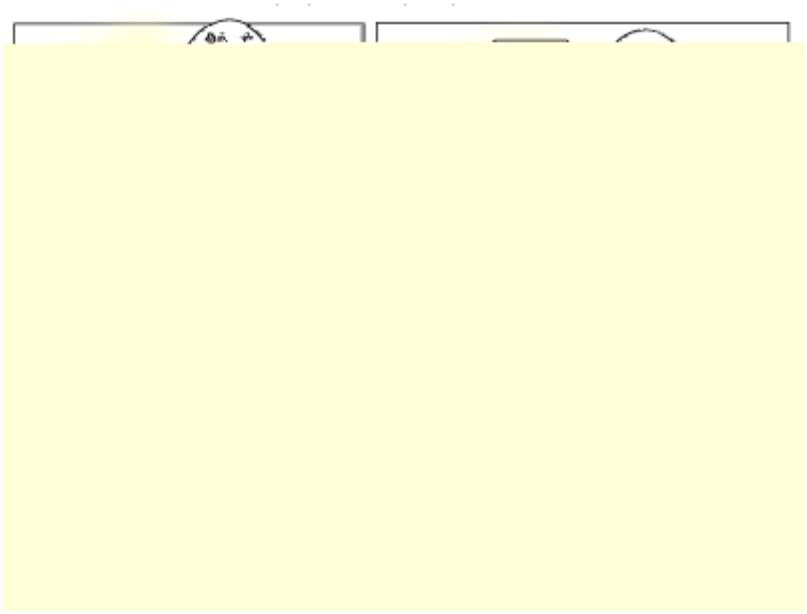

ロボットキャラを作ってみたよ。
トートツデスネ。
自分でも絵日記続ける意味がわからなくてねぇ。
ソーデスカ。
これからどうすればいいと思う？
トリアエズ仕事ナド シテミテハイカガデスカ？
いや、せっかく描いたし。
テヌキ・・・イヤリサイクルデスカ。

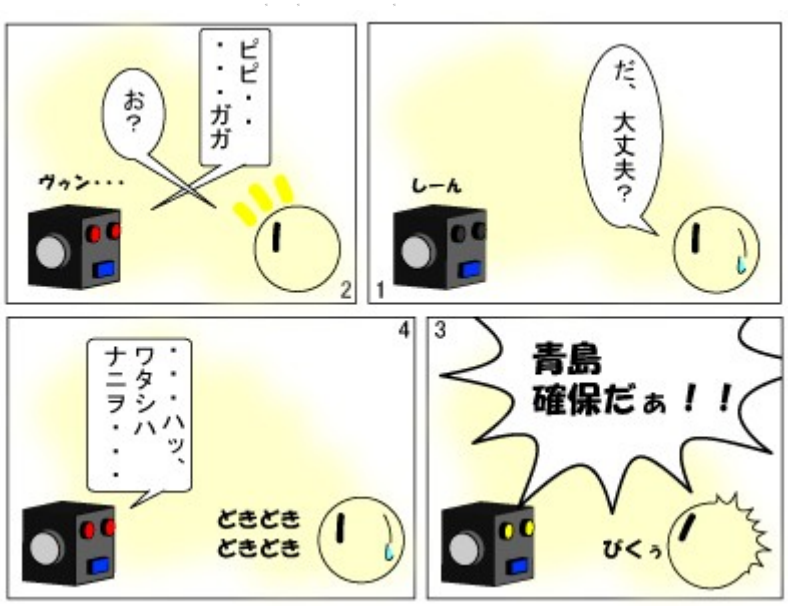

ぴる～
Bomb!
2
トクギ、デスカ？
ところでなにか特技とかあるの？
1
ほほう、そーでしたか
目カラ怪光線ナドダセマス。
3
ミサイルモダセマスガ。
いや、もういいです。
ずもっ
4

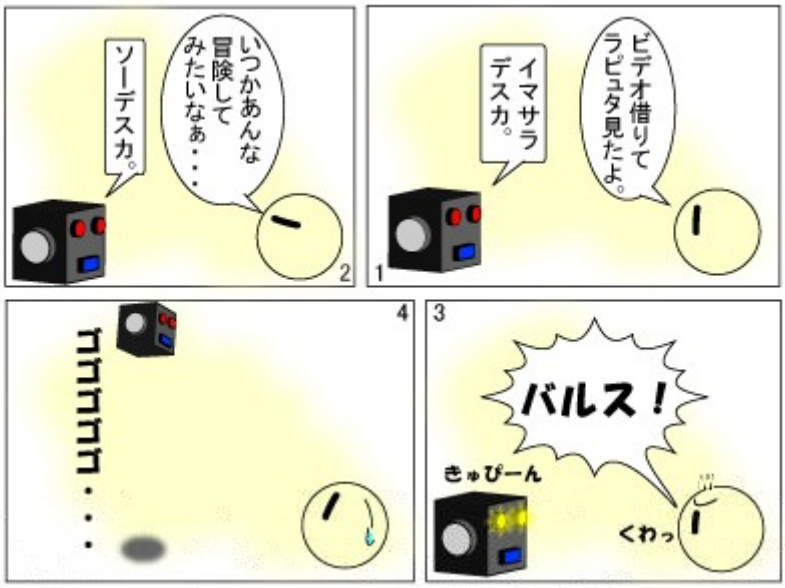

これからは
エコロジーだね
1
トートツデスネ
というわけで
改造するよ
2
ワタシヲデスカ?
・・・・・
3
改造したよ
どーよ?
4
ドーヨト
イワレマシテモ

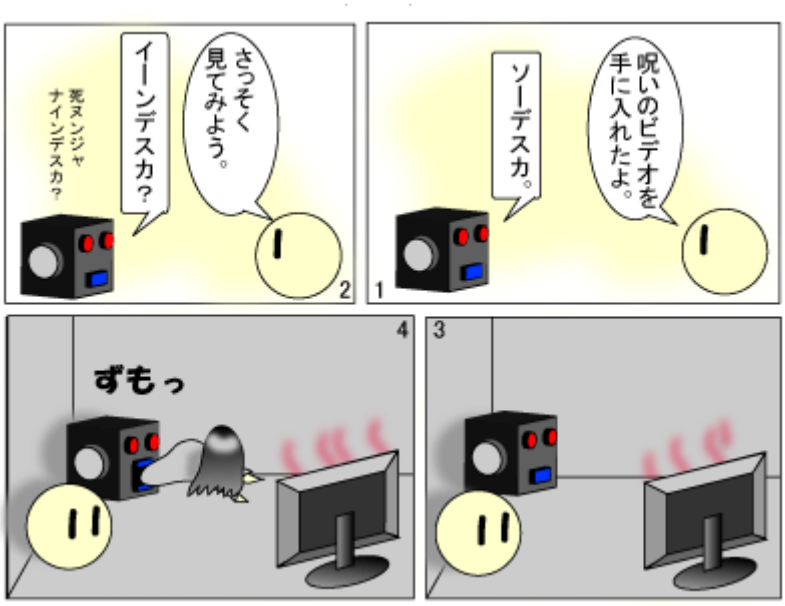

ナンデスカ？
じ～
1
いや～
なごむわぁ
ぽや～ん
ソーデスカ？
2

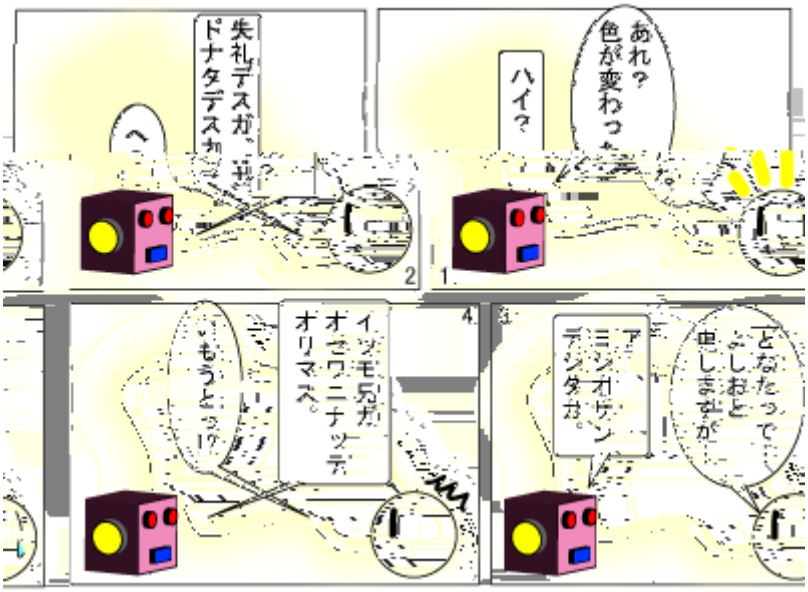

そーいや名前が無いね。いまさらだけど・・・
ソーデスネ。
うーん、どんなのがいいかなぁ
カッコイイノガイーデスネ。
ドーセナラ・・
ずもっ

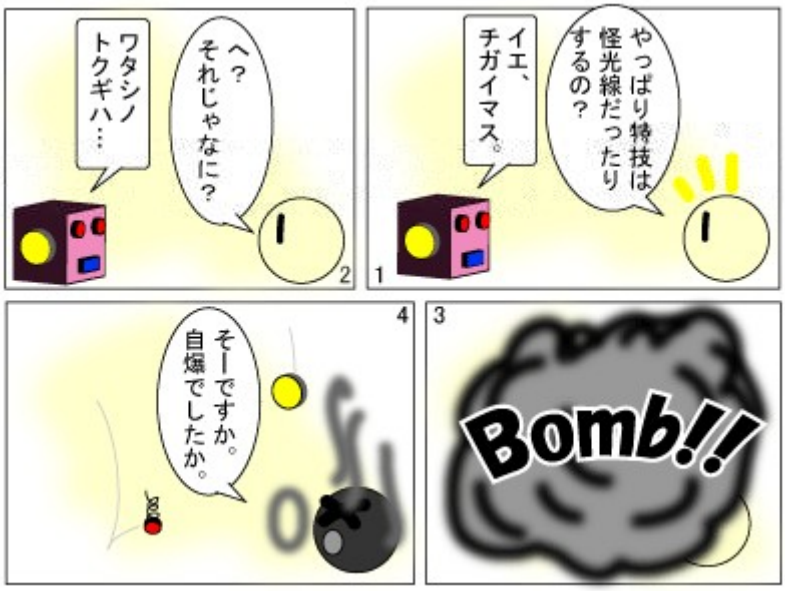


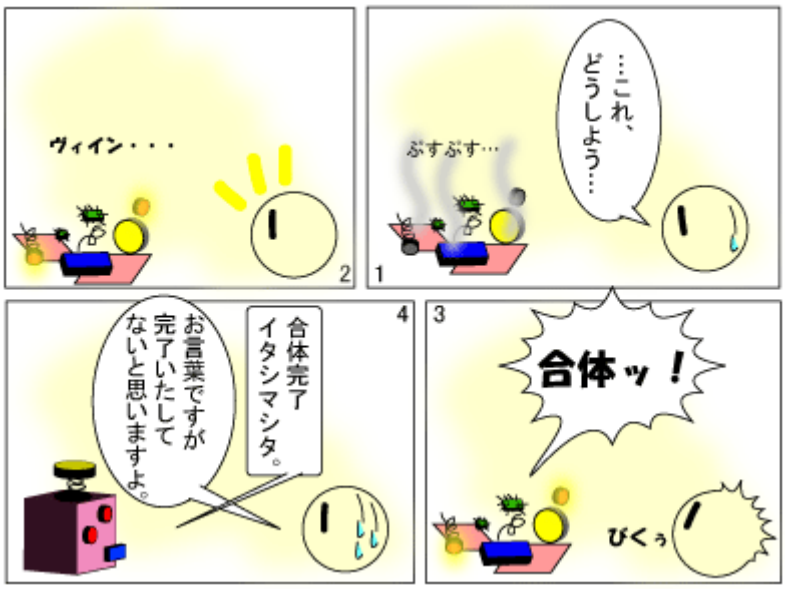

だ、大丈夫？
しーん
1
ピピ・・・ガガ
お？
ヴゥン・・・
2
青島
確保だぁ！！
びくぅ
3
・・・ハッ、
ワタシハ
ナニヲ・・・
どきどき
どきどき
4

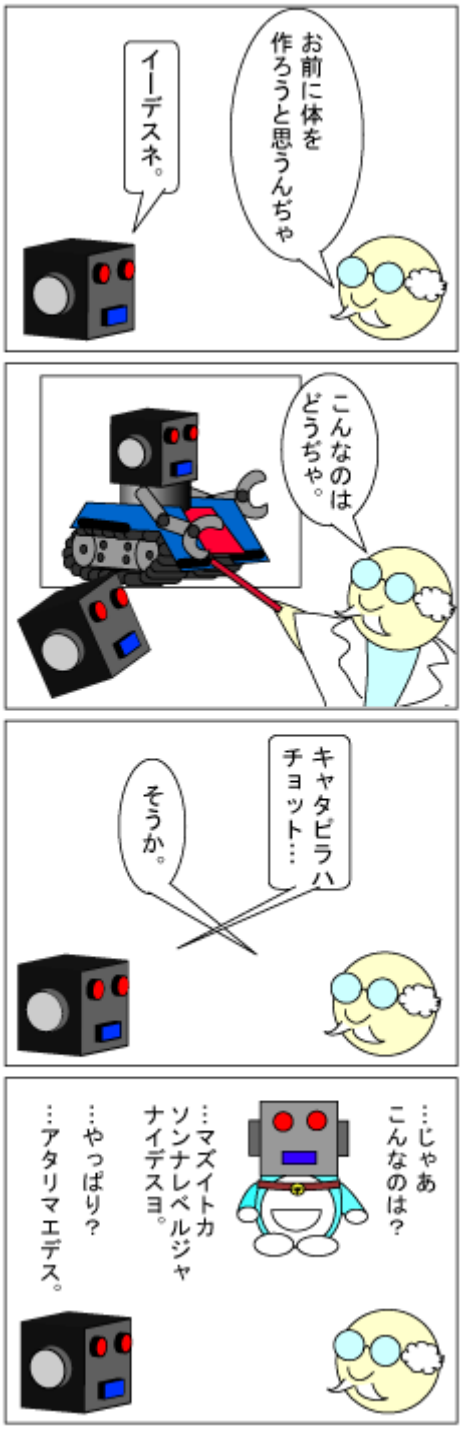

ビデオ借りてラピュタ見たよ。
イマサラデスカ。
1
いつかあんな冒険してみたいなぁ・・・
ソーデスカ。
2
バルス！
きゅぴーん
くわっ
3
ゴゴゴゴゴ・・・
4

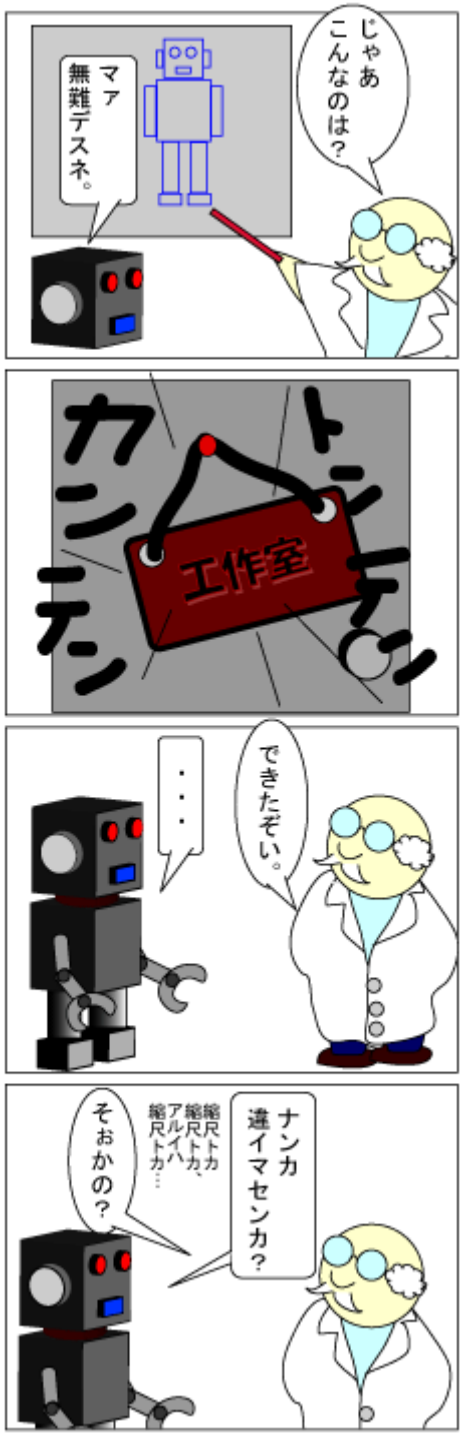

呪いのビデオを手に入れたよ。
ソーデスカ。
1
さっそく見てみよう。
イーンデスカ？
死ヌンジャナインデスカ？
2
3
ずもっ
4

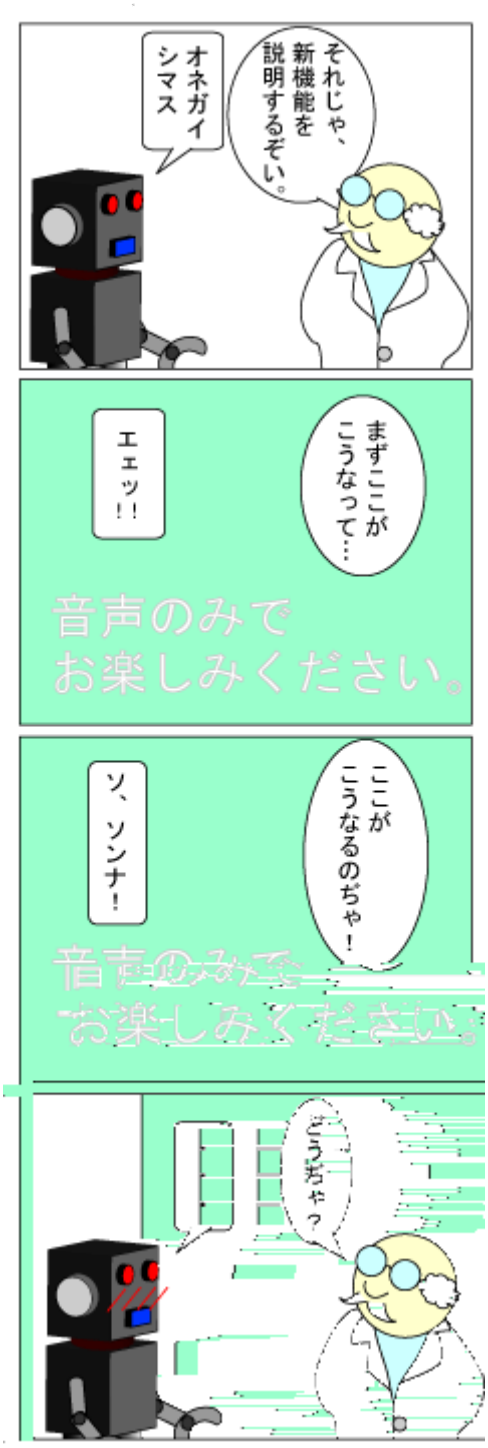

あれ？
色が変わっ
ハイ？
1
失礼デスガ。
ドナタデスカ
へ
2
となたって
ふしおと
申しますが
アミシオサン
デンタカ。
4
イツモ兄ガ
オセワニナッテ
オリマス。
もうとっ!?

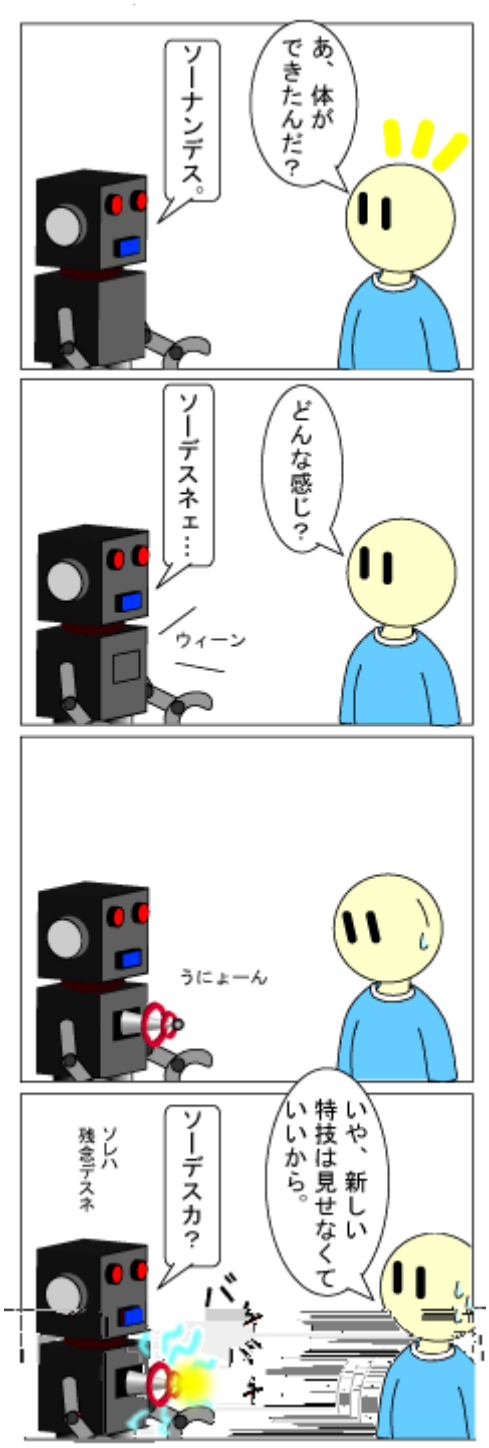

やっぱり特技は怪光線だったりするの？
イエ、チガイマス。
1
へ？それじゃなに？
ワタシノトクギハ…
2
Bomb!!
3
そーですか。自爆でしたか。
4

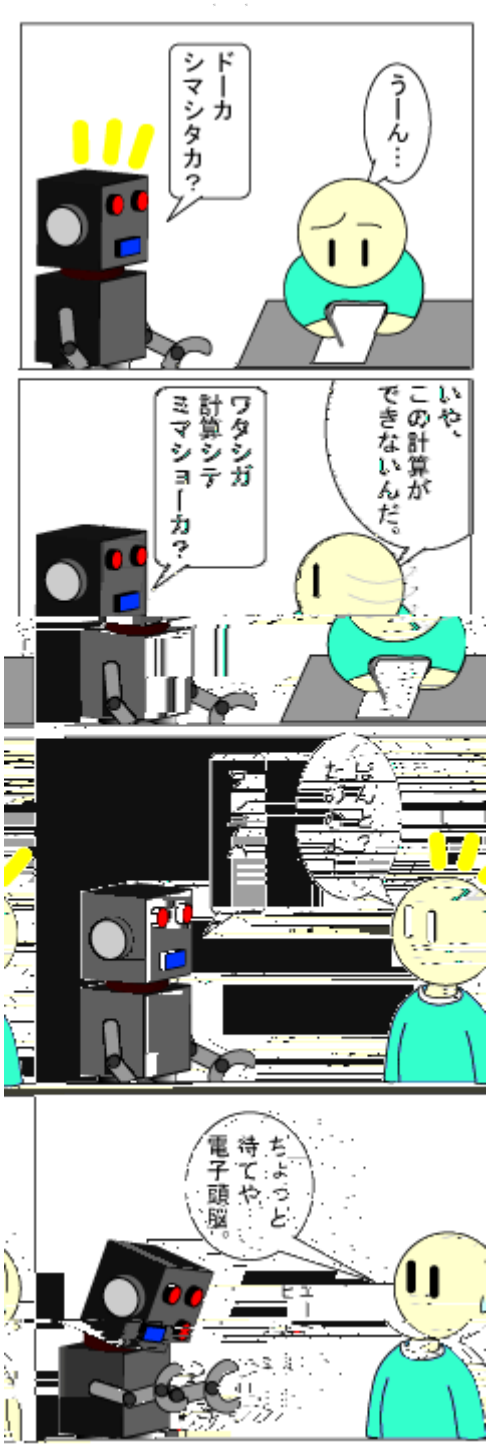

…これ、どうしよう…
ぶすぶす…
1
ヴィイン・・・
2
合体ッ！
3
ぴくぅ
合体完了イタシマシタ。
お言葉ですが完了いたしてないと思いますよ。
4

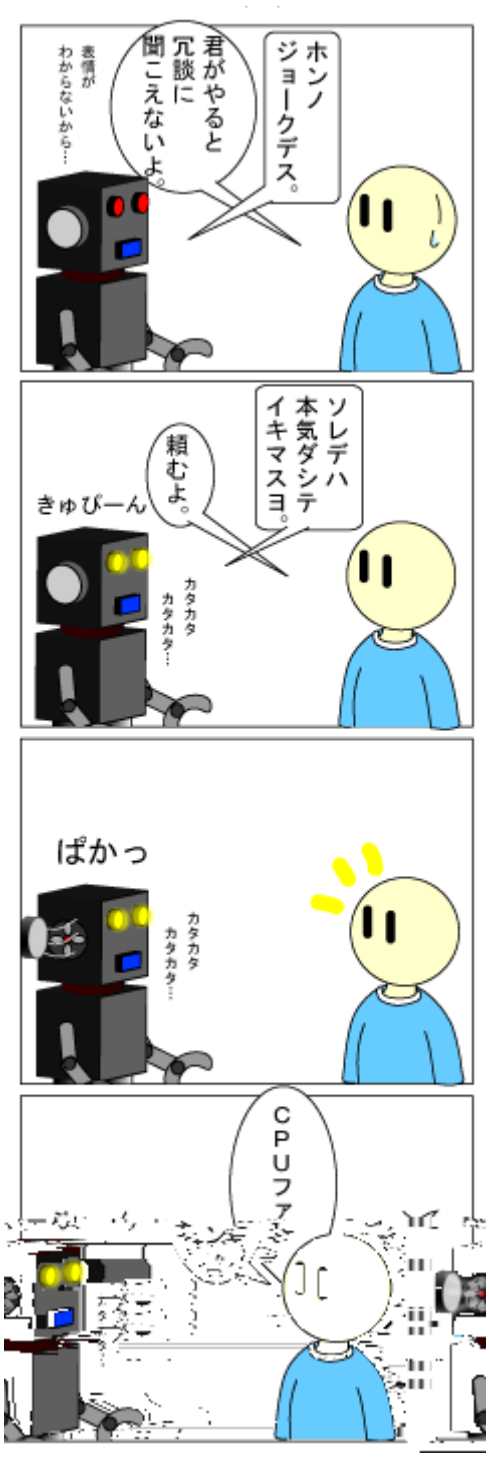

お前に体を作ろうと思うんぢゃ
イーデスネ。
こんなのはどうぢゃ。
キャタピラハチョット…
そうか。
…じゃあこんなのは？
…マズイトカソンナレベルジャナイデスヨ。
…やっぱり？
…アタリマエデス。


じゃあこんなのは？
マァ無難デスネ。
カンテン
工作室
できたぞい。
・・・
ナンカ違イマセンカ？
縮尺トカ縮尺トカ、アルイハ縮尺トカ…
そぉかの？

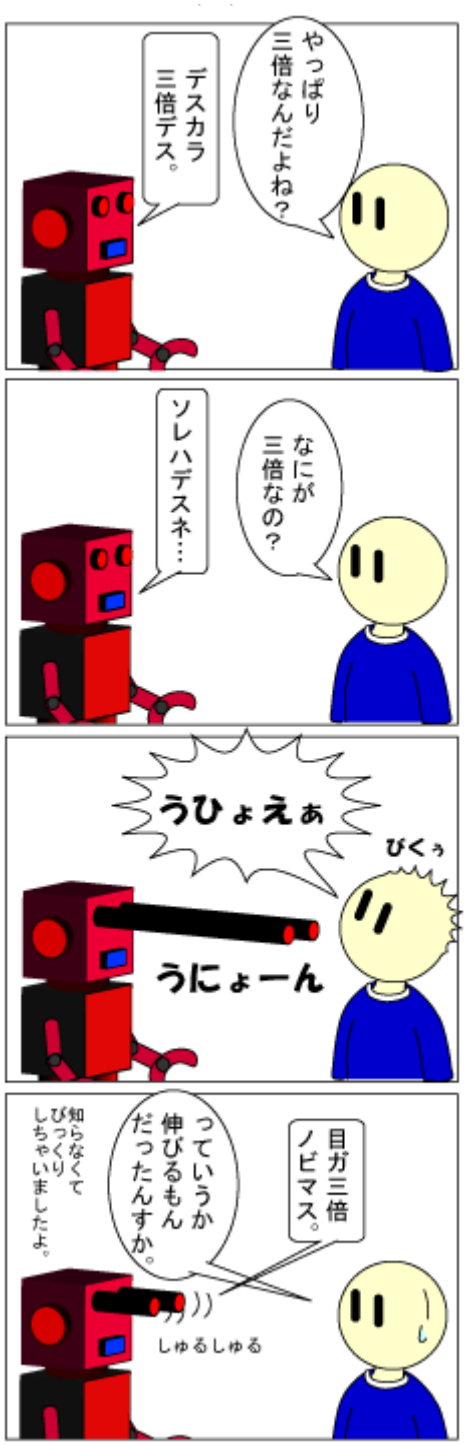

それじゃ、新機能を説明するぞい。
オネガイシマス
まずここがこうなって…
エェッ!!
音声のみでお楽しみください。
ここがこうなるのぢゃ！
ソ、ソンナ！
音声のみでお楽しみください。
どうぢゃ？


あ、体ができたんだ？
ソーナンデス。
どんな感じ？
ソーデスネェ…
ウィーン
うにょーん
いや、新しい特技は見せなくていいから。
ソーデスカ？
ソレハ残念デスネ
バチバチ

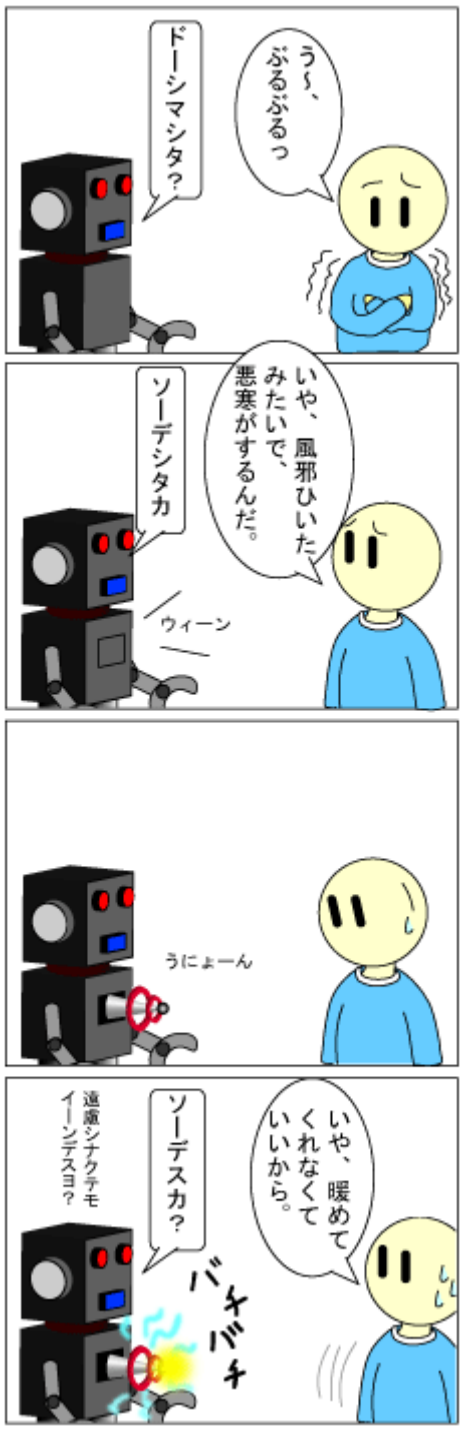

うーん…
ドーカシマシタカ？
いや、この計算ができないんだ。
ワタシガ計算シテミマショーカ？
ちょっと待てや電子頭脳。

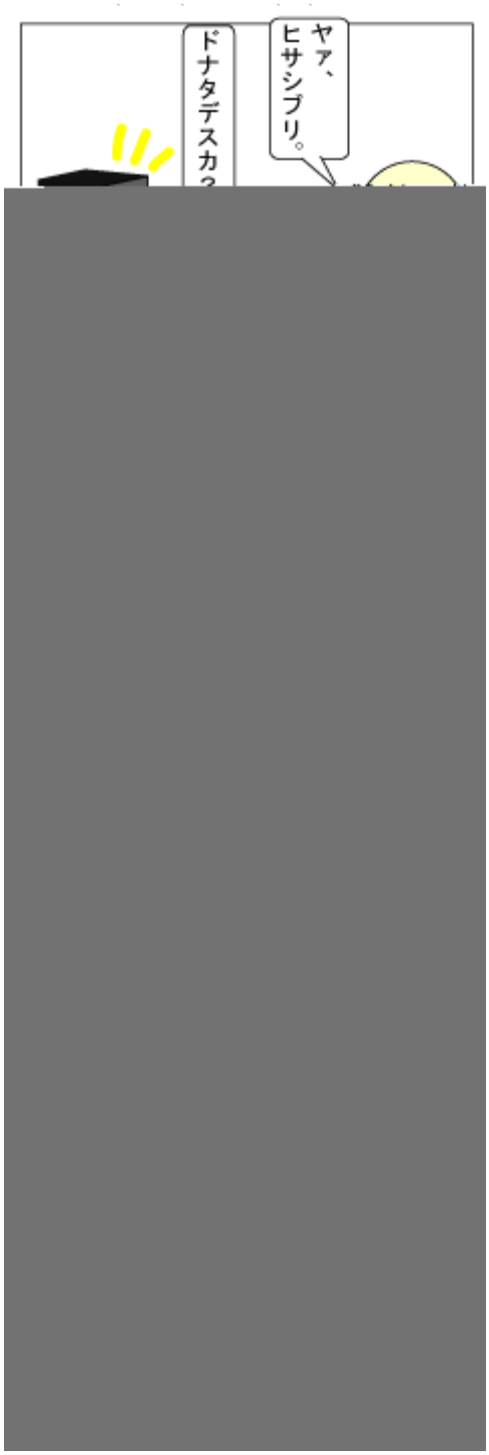

ホンノジョークデス。
君がやると冗談に聞こえないよ。
表情がわからないから…
ソレデハ本気ダシテイキマスヨ。
頼むよ。
きゅぴーん
カタカタカタカタ…
ぱかっ
カタカタカタカタ…
CPUファ


あれ？
えーと、
はじめまして？
イエ、
ワタシデス。
へー、
色変えたんだ。
ハイ、
イメチェンデス。
…やっぱり
三倍なの？
モチロン
三倍デス。

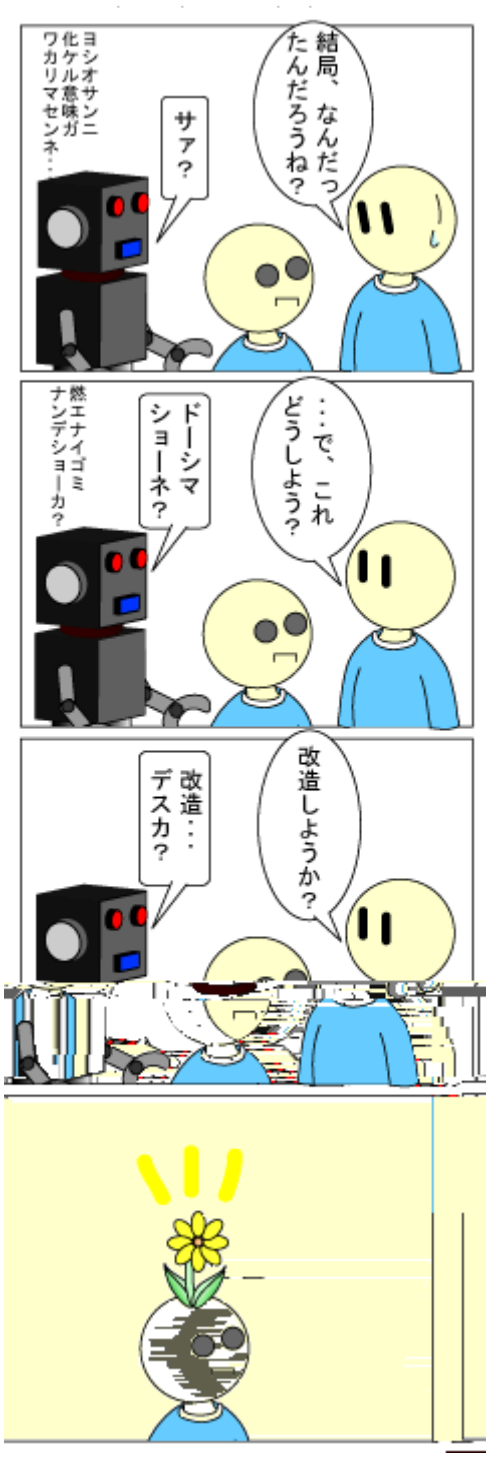

やっぱり三倍なんだよね？
デスカラ三倍デス。
なにが三倍なの？
ソレハデスネ…
うひょえぁ
びくぅ
うにょーん
目ガ三倍ノビマス。
っていうか伸びるもんだったんすか。
知らなくてびっくりしちゃいましたよ。
しゅるしゅる

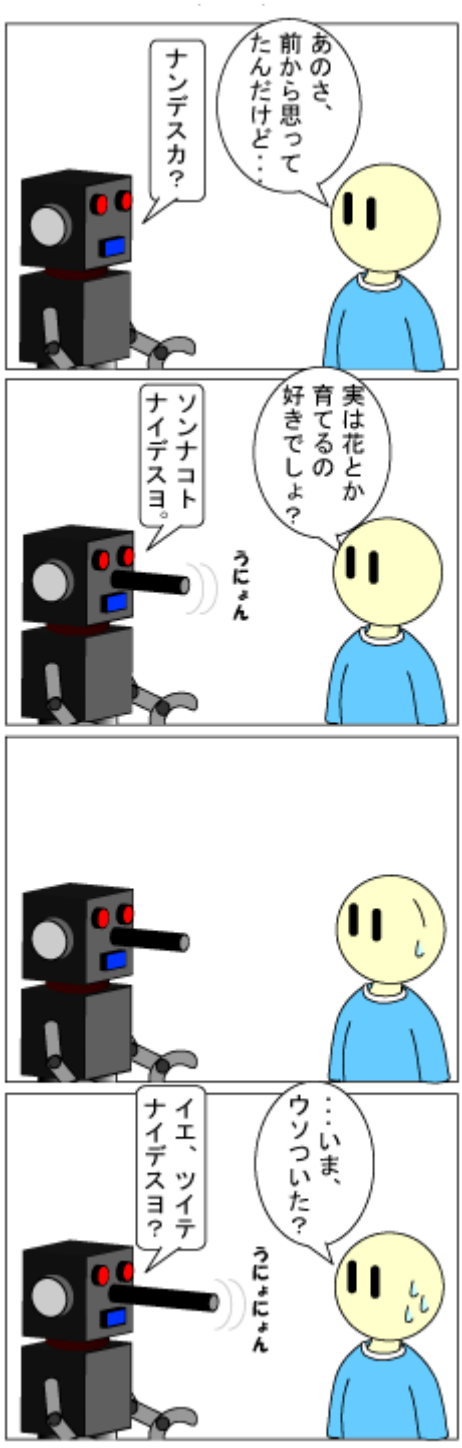

あ、お弁当が冷めちゃった
温メマショーカ？
へ？そんなことできるの？
ハイ
ウィーン
シュバババ
遠距離電子レンジ!?
ハイ ドーゾ

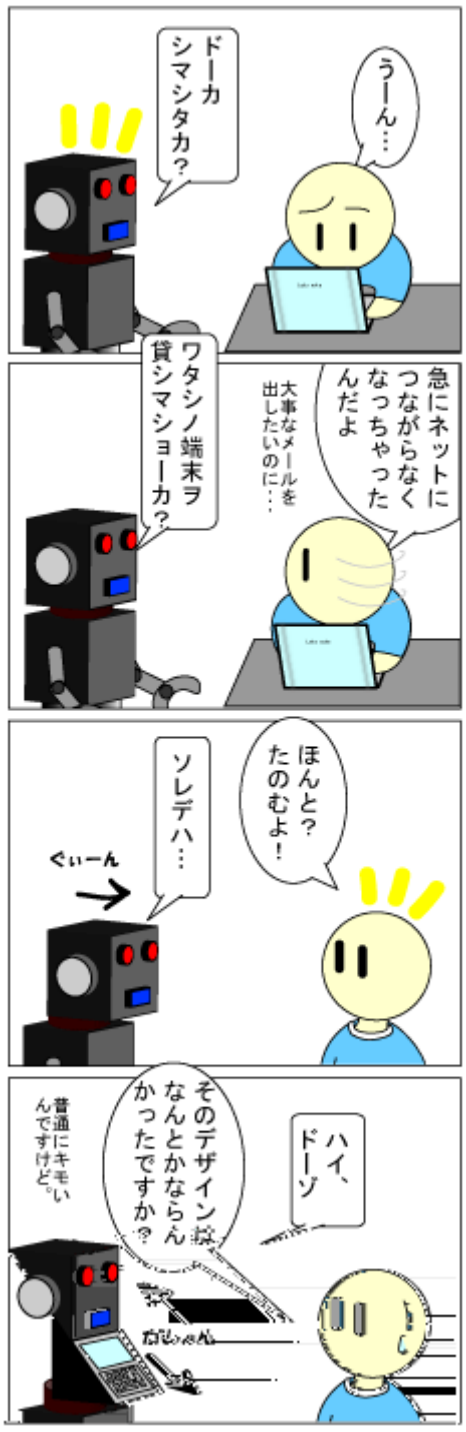

う～、ぶるぶるっ
ドーシマシタ？
いや、風邪ひいたみたいで、悪寒がするんだ。
ソーデシタカ
ウィーン
うにょーん
いや、暖めてくれなくていいから。
ソーデスカ？
バチバチ
遠慮シナクテモイーンデスヨ？

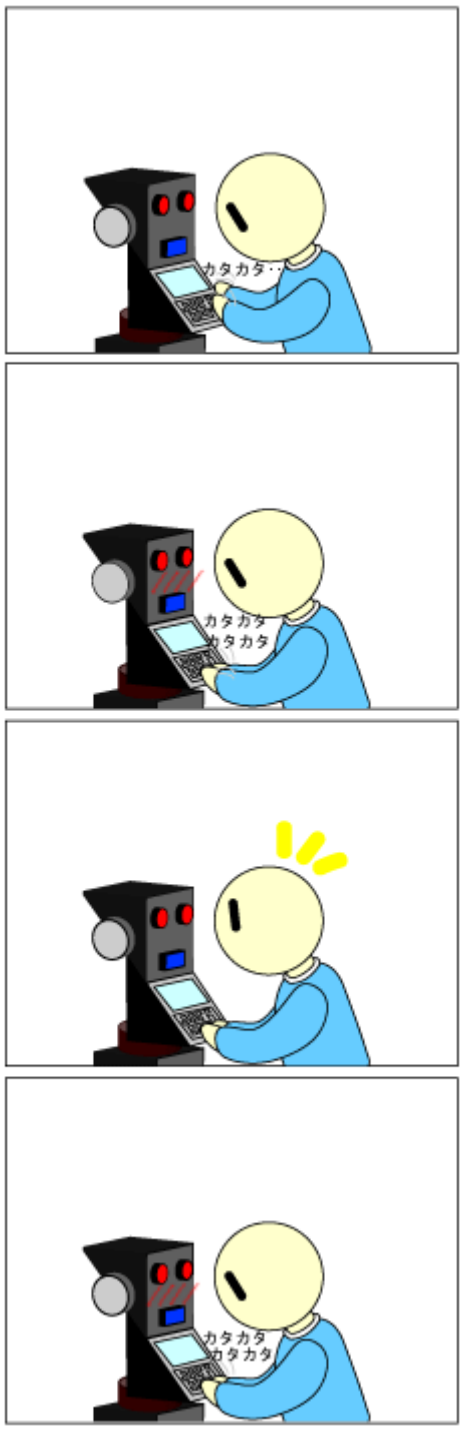

ヤァ、
ヒサシブリ。
ドナタデスカ？

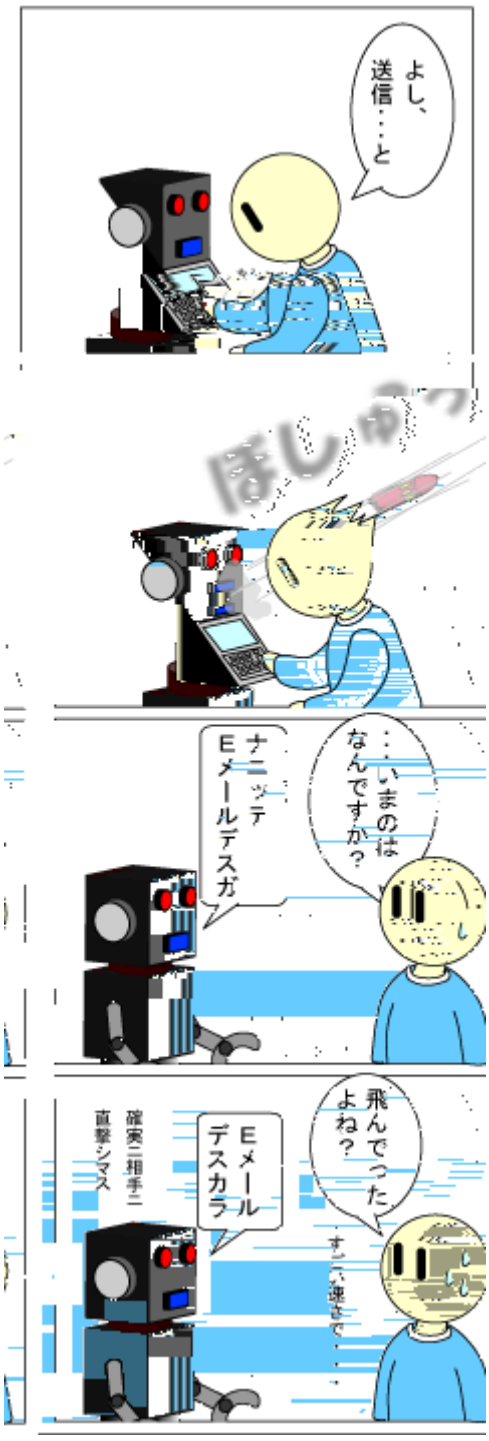

ム！本物ッ!!
ん？君は誰？
カクナル上ハ・・・・コレデモクラエッ !!
うわっ！
ジャキーン
ばしゃっ
あの～もしもし？
ぷすん・・・

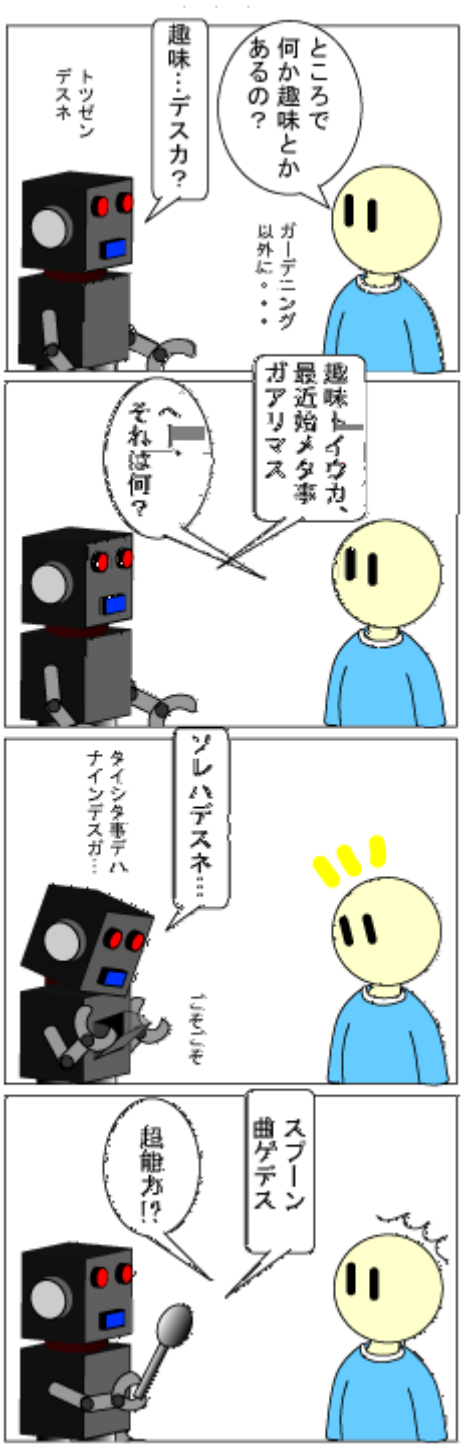

結局、なんだったんだろうね？
サァ？
ヨシオサンニ化ケル意味ガワカリマセンネ…
…で、これどうしよう？
ドーシマショーネ？
燃エナイゴミナンデショーカ？
改造しようか？
改造…デスカ？

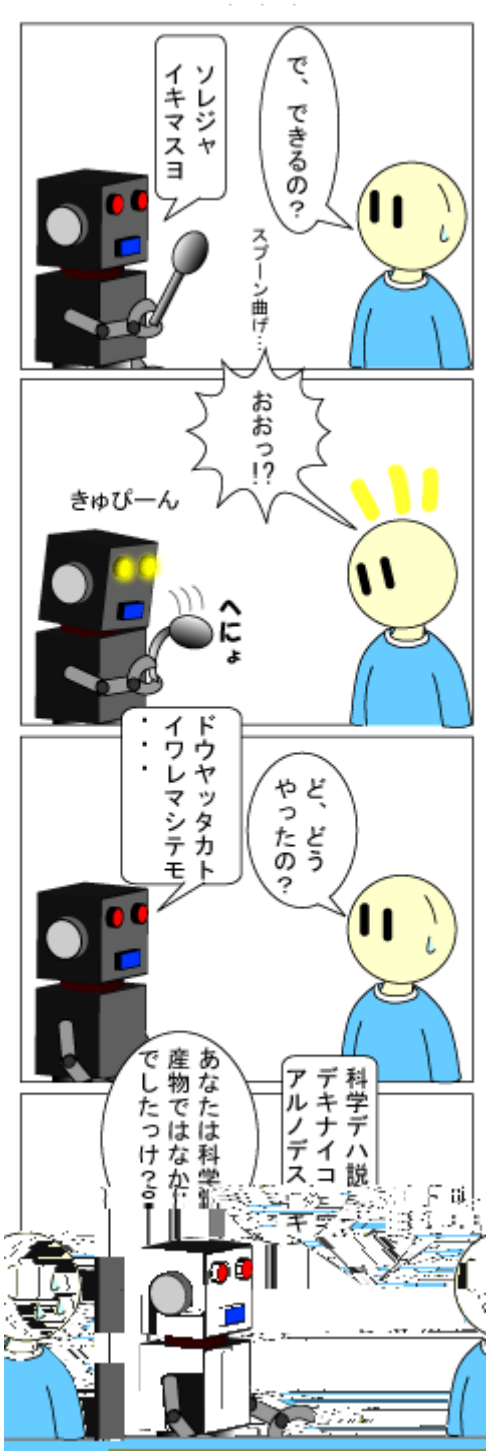

あのさ、前から思ってたんだけど…
ナンデスカ？
実は花とか育てるの好きでしょ？
うにょん
ソンナコトナイデスヨ。
…いま、ウソついた？
うにょにょん
イエ、ツイテナイデスヨ？

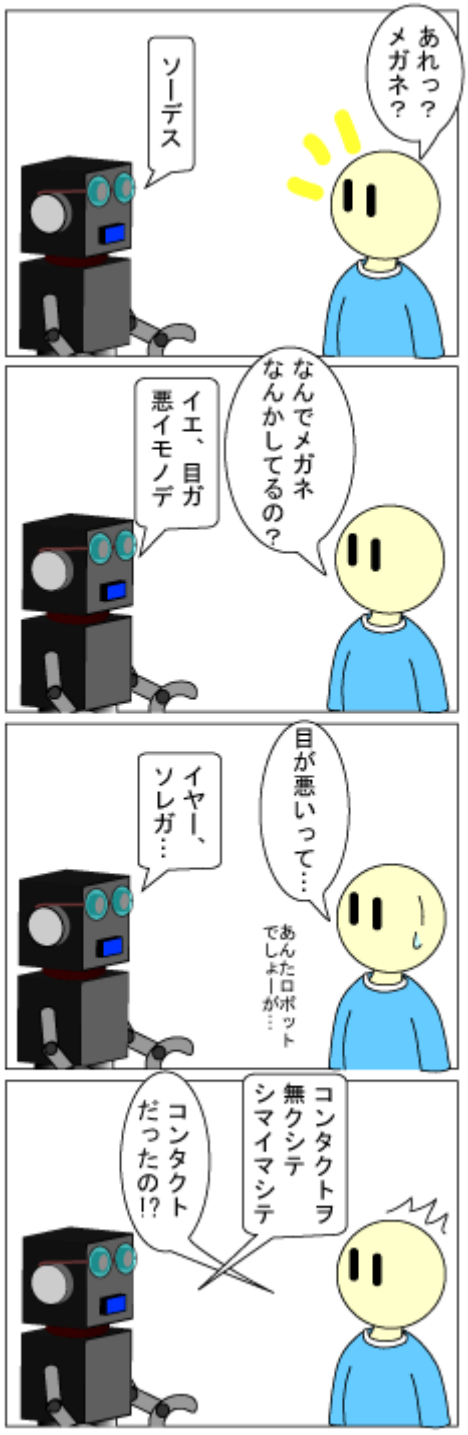

うーん…
ドーカシマシタカ？
急にネットにつながらなくなっちゃったんだよ
大事なメールを出したいのに…
ワタシノ端末ヲ貸シマショーカ？
ほんと？たのむよ！
ぐぃーん
ソレデハ…
ハイ、ドーゾ
そのデザイン採用なんとかならんかったですか？
普通にキモいんですけど。

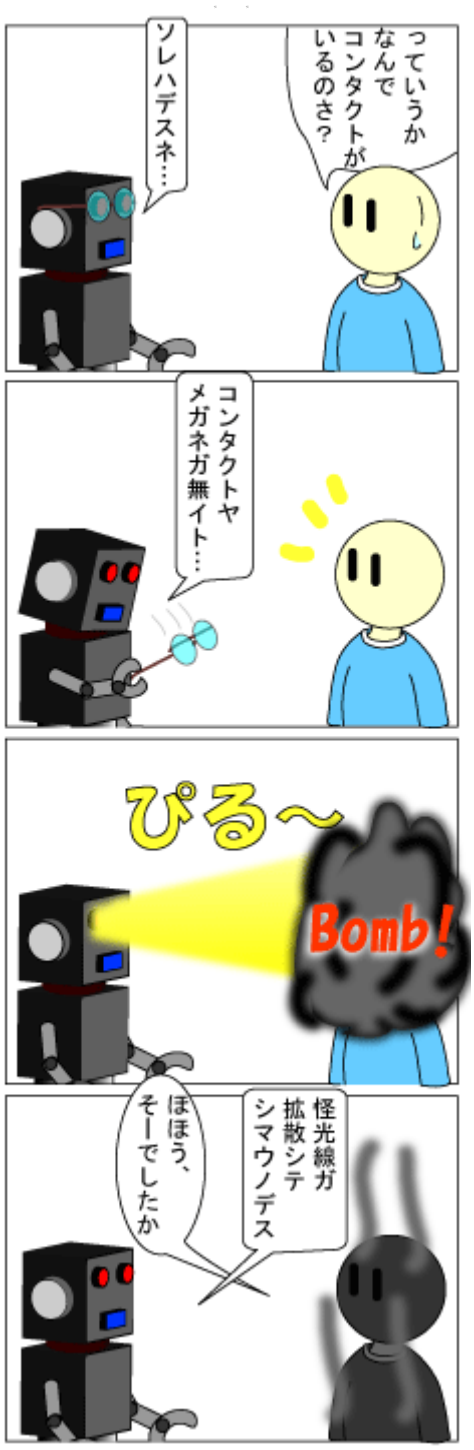

カタカタ…
カタカタ
カタカタ
カタカタ
カタカタ

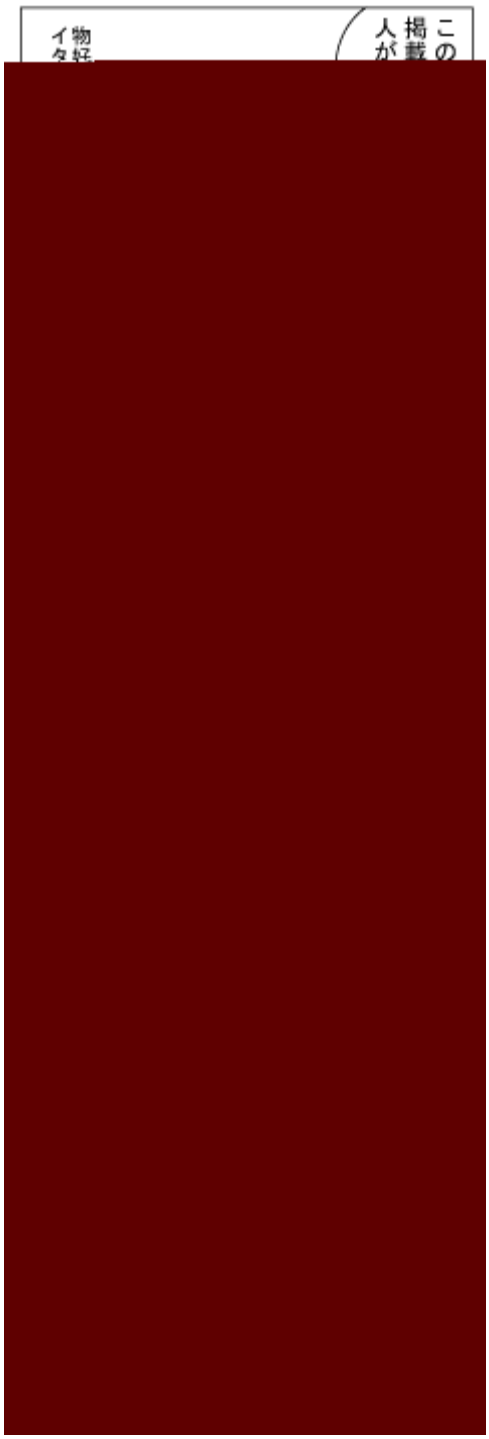

よし、送信…と
ぽしゅっ
…いまのはなんですか？
ナニッテEメールデスガ
飛んでったよね？
すごい速さで…
Eメールデスカラ
確実ニ相手ニ直撃シマス

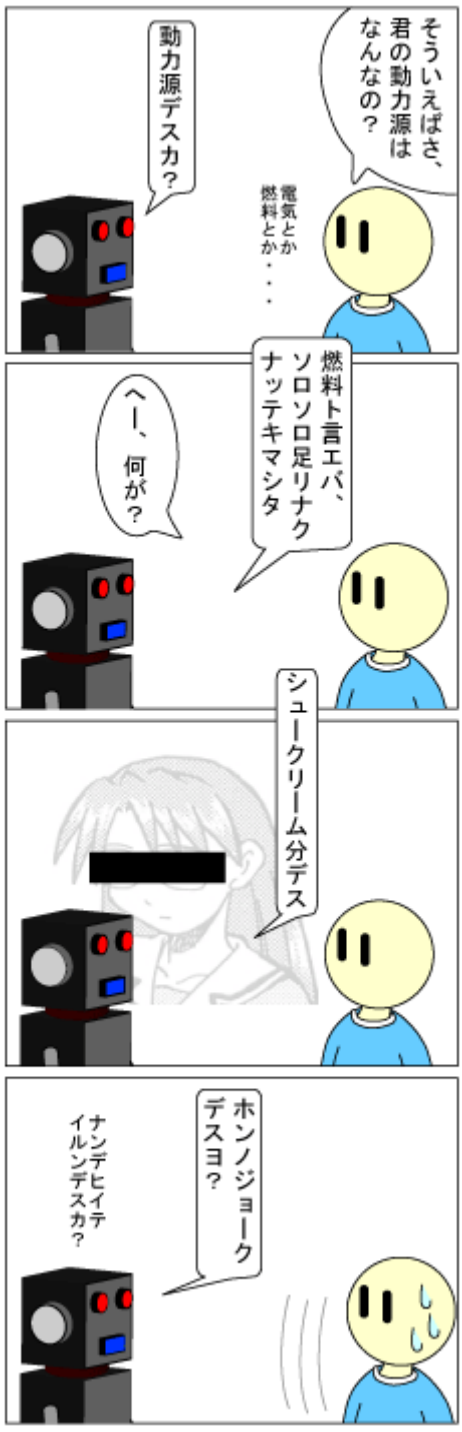

ところで何か趣味とかあるの？
ガーデニング以外に・・・
趣味…デスカ？
トツゼンデスネ
趣味トイウカ、最近始メタ事ガアリマス
へー、それは何？
ソレハデスネ…
ココニモ
タイシタ事デハナインデスガ…
スプーン曲ゲデス
超能力!?

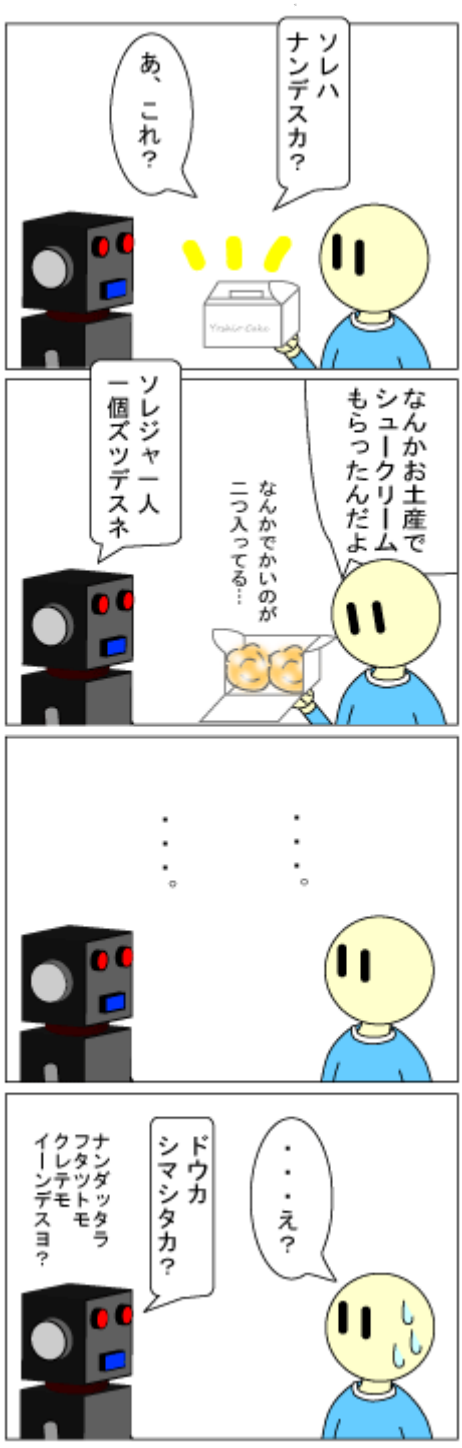

で、できるの？
ソレジャイキマスヨ
スプーン曲げ…
おおっ!?
きゅぴーん
へにょ
ど、どうやったの？
ドウヤッタカイワレマシテモ・・・
科学デハ説明デキナイコトデアルノデス
あなたは科学の産物ではなかったっけ？

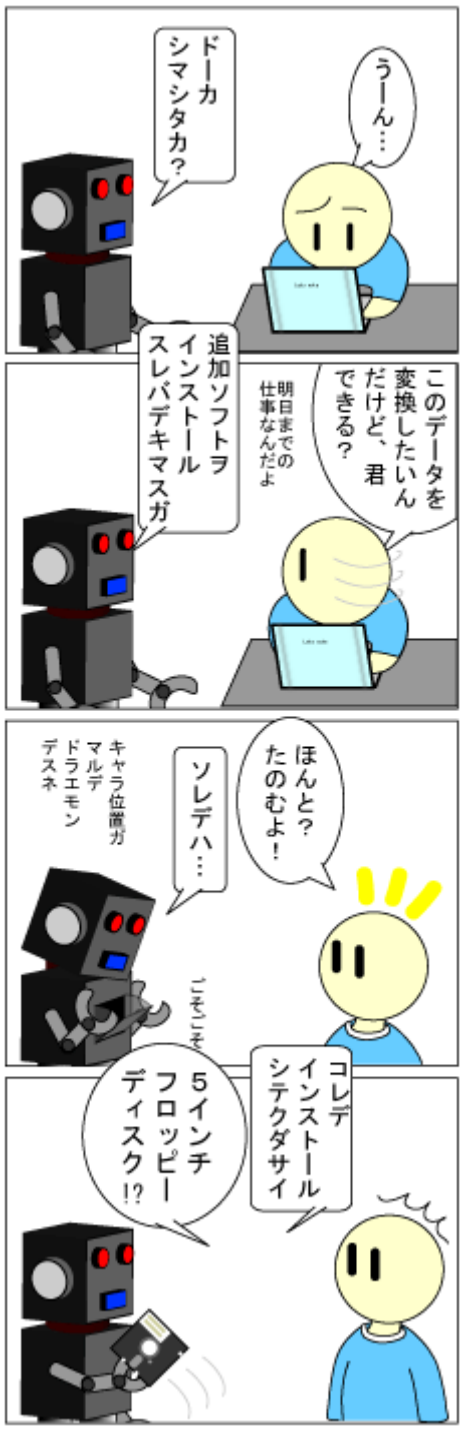

あれっ？メガネ？
ソーデス
なんでメガネなんかしてるの？
イエ、目ガ悪イモノデ
目が悪いって…
あんたロボットでしょーが…
イヤー、ソレガ…
コンタクトヲ無クシテシマイマシテ
コンタクトだったの!?

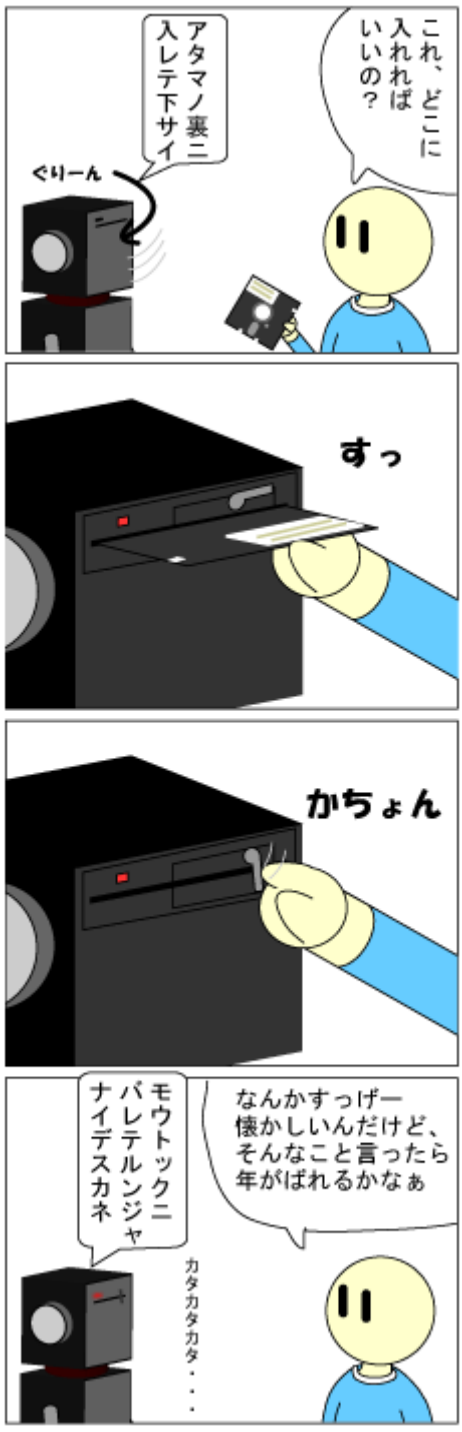

っていうか
なんで
コンタクトが
いるのさ？
ソレハデスネ…
コンタクトヤ
メガネガ無イト…
ぴる～
Bomb!
怪光線ガ
拡散シテ
シマウノデス
ほほう、
そーでしたか


この
掲載
人が
物好
イタ

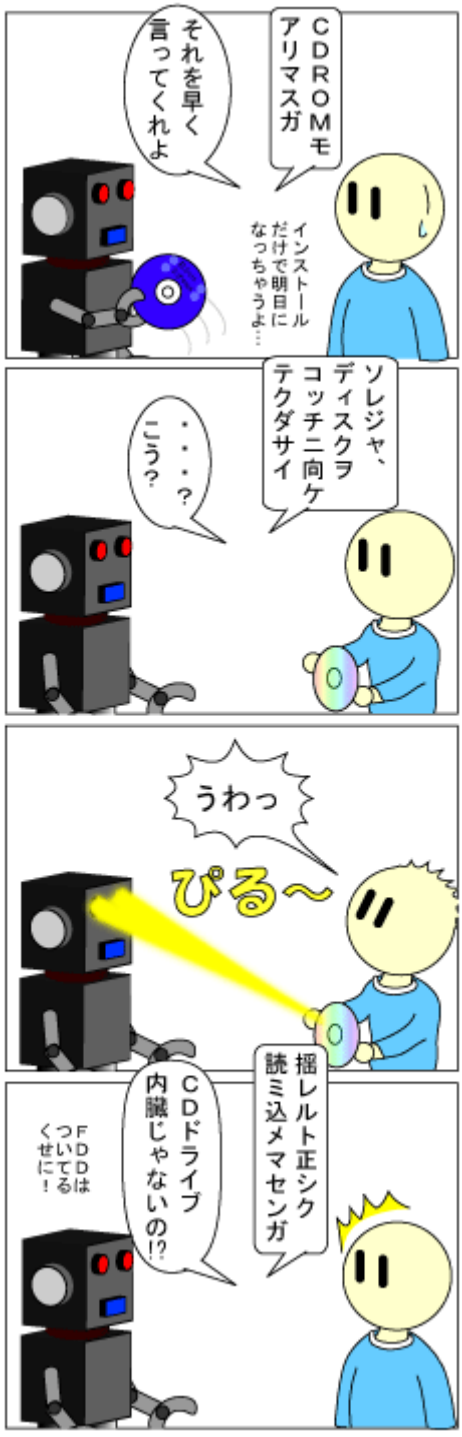

そういえばさ、君の動力源はなんなの？
電気とか燃料とか・・・
動力源デスカ？
燃料ト言エバ、ソロソロ足リナクナッテキマシタ
へー、何が？
シュークリーム分デス
ホンノジョークデスヨ？
ナンデヒイテイルンデスカ？


ソレハ
ナンデスカ？
あ、これ？
なんかお土産で
シュークリーム
もらったんだよ
なんかでかいのが
二つ入ってる…
ソレジャ一人
一個ズツデスネ
・・・。
・・・。
・・・え？
ドウカ
シマシタカ？
ナンダッタラ
フタツトモ
クレテモ
イーンデスヨ？


うーん…
ドーカシマシタカ？
このデータを変換したいんだけど、君できる？
明日までの仕事なんだよ
追加ソフトヲインストールスレバデキマスガ
ほんと？たのむよ！
ソレデハ…
キャラ位置ガマルデドラエモンデスネ
ごそごそ
コレデインストールシテクダサイ
5インチフロッピーディスク!?


これ、どこに入れればいいの？
アタマノ裏ニ入レテ下サイ
ぐりーん
すっ
かちょん
なんかすっげー懐かしいんだけど、そんなこと言ったら年がばれるかなぁ
モウトックニパレテルンジャナイデスカネ
カタカタカタ・・・


かたかた
かたかた・・・
ディスク2ヲイレテクダサイ
へいへい
かたっ
すっ
ソーデスネ、デフォルトインストールデイイナラ…
ところでこのソフト、何枚組み？
カタカタカタ・・・
５０枚デスネ
５０枚!?
約５０ＭＢデスカラ…


CDROMモアリマスガ
インストールだけで明日になっちゃうよ…
それを早く言ってくれよ
ソレジャ、ディスクヲコッチニ向ケテクダサイ
・・・？こう？
うわっ
ぴる～
揺レルト正シク読ミ込メマセンガ
CDドライブ内臓じゃないの!?
FDはついてるくせに！

挑戦 夫
挑

× 埋
３×３
候補
候補
易
候補
矛盾
帰 深

省
我 恐 尽蔵
渡
埋 埋 矛盾
途 埋 候補 軸 帰

```
#!/usr/bin/ruby

$L = 9
$L2 = $L*$L

class Array
   def sum
      self.inject(0){|result, i| result + i }
   end
end

class Grid
  def initialize (l,a)
    @level = l
    @changed = false
    @data = Array.new(a)
    @flag = Array.new($L2){|i|
      Array.new($L){|j| 1}
    }
    killall
```

```
    puts "Recursive Level: #{@level}"
    determine
  end

  def rest
    return @data.select{|i| i== 0}.size
  end

  def solved
    return rest == 0
  end

  def showflag
    i = 0
    @flag.each{|f|
      x = i % $L
      y = (i/$L)
      i = i + 1
    }
  end

  def pos2index(x,y)
    return y*$L + x
  end

  def kill_xline(i,n)
    i = (i / $L)*$L
    for j in 0..$L-1
      @flag[i+j][n-1] = 0
    end
  end

  def kill_yline(i,n)
    i = (i % $L)
    for j in 0..$L-1
      @flag[i+j*$L][n-1] = 0
    end
  end

  def kill_block(i,n)
    x = (i % $L)
    y = (i / $L)
```

```
    x = (x / 3) * 3
    y = (y / 3) * 3
    i = pos2index(x,y)
    for j in 0..2
      @flag[i+j*$L][n-1] = 0
      @flag[i+j*$L+1][n-1] = 0
      @flag[i+j*$L+2][n-1] = 0
    end
  end

  def killflag(i,n)
    @flag[i].fill(0)
    kill_xline(i,n)
    kill_yline(i,n)
    kill_block(i,n)
  end

  def killall
    @data.size.times{|i|
      killflag(i,@data[i]) if @data[i] != 0
    }
  end

  def place_number(i,n)
    return if @data[i] != 0
    x = (i % $L) +1
    y = i/$L + 1
    @data[i] = n + 1
    @changed = true
    killflag(i,n+1)
  end

  def check(num)
    t = Array.new($L){|i| 0}
    num.each{|i|
      $L.times{|n|
        t[n] = t[n] + @flag[i][n]
      }
    }
    while n = t.index(1)
      num.each{|i|
        if @flag[i][n] == 1
```

```
          place_number(i,n)
        end
        t[n] = 0
      }
    end
  end

  def check_line
    $L.times{|y|
      check(Array.new($L){|i| y*$L + i})
    }
    $L.times{|x|
      check(Array.new($L){|i| i*$L + x})
    }
    $L.times{|n|
      x = (n%3)*3
      y = (n/3)*3
      check(Array.new($L){|i| (i%3 + x) + (i/3 + y)*$L})
    }
  end

  def determine
    while 1
      @changed = false
      check_line
      return if @changed == false
    end
  end

  def solve_recursive(i,n)
    d = @data
    d[i] = n
    g = Grid.new(@level+1,d)
    return g.solve
  end

  def solve

    if solved
      puts "----------------------------"
      puts "Solved! at Level #{@level}"
      show
```

```
      exit 1
      return 1
    end

    min_index = 0
    min_value = 10
    @data.size.times{|i|
      if @data[i] == 0
        s = @flag[i].sum
        if min_value > s
          min_index = i
          min_value = s
        end
      end
    }
    if min_value == 0
      return 0
    end
    t = @flag[min_index]
    c = Array.new
    while n = t.index(1)
      c.push n+1
      t[n] = 0
    end
    x = min_index % $L +1
    y = min_index / $L + 1
    ans = 0
    c.each{|n|
      ans = ans + solve_recursive(min_index,n)
    }
    return ans
  end

  def show
      puts "----------------------------"
    $L.times{|i|
      j = $L*i
      k = j + $L-1
      puts @data[j..k].join(",")
    }
      puts "----------------------------"
  end
```

```
end

a = Array.new
while line=gets
  line.chomp.split(/,/).each{|i|
    a.push i.to_i
  }
end
g = Grid.new(0,a)
ans = g.solve

puts "This quenstion has #{ans} answer[s]."
```

与 V 与

```
0,0,6,0,0,0,0,0,1
0,7,0,0,6,0,0,5,0
8,0,0,1,0,3,2,0,0
0,0,5,0,4,0,8,0,0
0,4,0,7,0,2,0,9,0
0,0,8,0,1,0,7,0,0
0,0,1,2,0,5,0,0,3
0,6,0,0,7,0,0,8,0
2,0,0,0,0,0,4,0,0
```

与

```
$ ruby sudoku.rb sample1.dat
Recursive Level: 0
----------------------------
Solved! at Level 0
----------------------------
5,3,6,8,2,7,9,4,1
1,7,2,9,6,4,3,5,8
8,9,4,1,5,3,2,6,7
7,1,5,3,4,9,8,2,6
6,4,3,7,8,2,1,9,5
9,2,8,5,1,6,7,3,4
4,8,1,2,9,5,6,7,3
3,6,9,4,7,1,5,8,2
2,5,7,6,3,8,4,1,9
----------------------------
```

帰 帰
誌

```
9,0,0,0,2,3,0,0,8
0,0,7,0,0,0,0,9,0
0,0,6,4,0,0,0,0,0
0,0,5,2,0,0,0,0,3
0,8,0,0,0,0,0,6,0
2,0,0,0,0,5,7,0,0
0,0,0,0,0,6,3,0,0
0,5,0,0,0,0,4,0,0
0,0,0,8,7,0,0,0,6
```

与

```
$ ruby sudoku.rb sample2.dat
Recursive Level: 0
Recursive Level: 1
Recursive Level: 2
Recursive Level: 3
Recursive Level: 1
Recursive Level: 2
Recursive Level: 3
---------------------------
Solved! at Level 3
---------------------------
9,4,1,5,2,3,6,7,8
5,3,7,6,1,8,2,9,4
8,2,6,4,9,7,1,3,5
7,9,5,2,6,1,8,4,3
1,8,3,7,4,9,5,6,2
2,6,4,3,8,5,7,1,9
4,7,8,9,5,6,3,2,1
6,5,9,1,3,2,4,8,7
3,1,2,8,7,4,9,5,6
---------------------------
```

帰 帰
帰

```
0,0,5,3,0,0,0,0,0
8,0,0,0,0,0,0,2,0
0,7,0,0,1,0,5,0,0
4,0,0,0,0,5,3,0,0
0,1,0,0,7,0,0,0,6
0,0,3,2,0,0,0,8,0
0,6,0,5,0,0,0,0,9
0,0,4,0,0,0,0,3,0
0,0,0,0,0,9,7,0,0

$ $ ruby sudoku.rb hardest.dat
Recursive Level: 0
(中略)
Recursive Level: 7
---------------------------
Solved! at Level 7
---------------------------
1,4,5,3,2,7,6,9,8
8,3,9,6,5,4,1,2,7
6,7,2,9,1,8,5,4,3
4,9,6,1,8,5,3,7,2
2,1,8,4,7,3,9,5,6
7,5,3,2,9,6,4,8,1
3,6,7,5,4,2,8,1,9
9,8,4,7,6,1,2,3,5
5,2,1,8,3,9,7,6,4
---------------------------
```

帰 7 ケ

秒

帰 枝

至

& 開

```
$ gem install ruby-graphviz
```

```
require 'rubygems'
require 'graphviz'
```

q 忘

```
#!/usr/bin/ruby
require 'rubygems'
require 'graphviz'

$L = 9
$L2 = $L*$L

$graph = GraphViz::new("G", { :type => "digraph", :use => "dot", :output =>
"png", :file => "sudoku.png"})

class Array
   def sum
      self.inject(0){|result, i| result + i }
   end
end

class Grid
  def initialize (l,a)
    @level = l
    @changed = false
    @data = Array.new(a)
    @flag = Array.new($L2){|i|
      Array.new($L){|j| 1}
    }
    killall
    puts "Recursive Level: #{@level}"

    determine
    s = ""

    $L.times{|i|
      j = $L*i
      k = j + $L-1
```

```
      s = s + @data[j..k].join(" ")
      s = s + "¥n"
    }
    @node = $graph.add_node(s)
  end

  def get_node
    return @node
  end

  def rest
    return @data.select{|i| i== 0}.size
  end

  def solved
    return rest == 0
  end

  def showflag
    i = 0
    @flag.each{|f|
      x = i % $L
      y = (i/$L)
      i = i + 1
    }
  end

  def pos2index(x,y)
    return y*$L + x
  end

  def kill_xline(i,n)
    i = (i / $L)*$L
    for j in 0..$L-1
      @flag[i+j][n-1] = 0
    end
  end

  def kill_yline(i,n)
    i = (i % $L)
    for j in 0..$L-1
      @flag[i+j*$L][n-1] = 0
```

```
    end
  end

  def kill_block(i,n)
    x = (i % $L)
    y = (i / $L)
    x = (x / 3) * 3
    y = (y / 3) * 3
    i = pos2index(x,y)
    for j in 0..2
      @flag[i+j*$L][n-1] = 0
      @flag[i+j*$L+1][n-1] = 0
      @flag[i+j*$L+2][n-1] = 0
    end
  end

  def killflag(i,n)
    @flag[i].fill(0)
    kill_xline(i,n)
    kill_yline(i,n)
    kill_block(i,n)
  end

  def killall
    @data.size.times{|i|
      killflag(i,@data[i]) if @data[i] != 0
    }
  end

  def place_number(i,n)
    return if @data[i] != 0
    x = (i % $L) +1
    y = i/$L + 1
    @data[i] = n + 1
    @changed = true
    killflag(i,n+1)
  end

  def check(num)
    t = Array.new($L){|i| 0}
    num.each{|i|
      $L.times{|n|
```

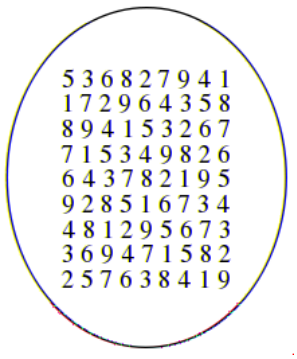
```
        t[n] = t[n] + @flag[i][n]
      }
    }
    while n = t.index(1)
      num.each{|i|
        if @flag[i][n] == 1
          place_number(i,n)
        end
        t[n] = 0
      }
    end
  end

  def check_line
    $L.times{|y|
      check(Array.new($L){|i| y*$L + i})
    }
    $L.times{|x|
      check(Array.new($L){|i| i*$L + x})
    }
    $L.times{|n|
      x = (n%3)*3
      y = (n/3)*3
      check(Array.new($L){|i| (i%3 + x) + (i/3 + y)*$L})
    }
  end

  def determine
    while 1
      @changed = false
      check_line
      return if @changed == false
    end
  end

  def solve_recursive(i,n)
    d = @data
    d[i] = n
    g = Grid.new(@level+1,d)
    $graph.add_edge(@node, g.get_node)
    return g.solve
  end
```

```
  def solve

    if solved
      puts "----------------------------"
      puts "Solved! at Level #{@level}"
      show
      return 1
    end

    min_index = 0
    min_value = 10
    @data.size.times{|i|
      if @data[i] == 0
        s = @flag[i].sum
        if min_value > s
          min_index = i
          min_value = s
        end
      end
    }
    if min_value == 0
      return 0
    end
    t = @flag[min_index]
    c = Array.new
    while n = t.index(1)
      c.push n+1
      t[n] = 0
    end
    x = min_index % $L +1
    y = min_index / $L + 1
    ans = 0
    c.each{|n|
      ans = ans + solve_recursive(min_index,n)
    }
    return ans
  end

  def show
      puts "----------------------------"
    $L.times{|i|
```

```
      j = $L*i
      k = j + $L-1
      puts @data[j..k].join(",")
    }
      puts "----------------------------"
  end
end

a = Array.new
while line=gets
  line.chomp.split(/,/).each{|i|
    a.push i.to_i
  }
end
g = Grid.new(0,a)
ans = g.solve

puts "This quenstion has #{ans} answer[s]."
$graph.output()
```

帰

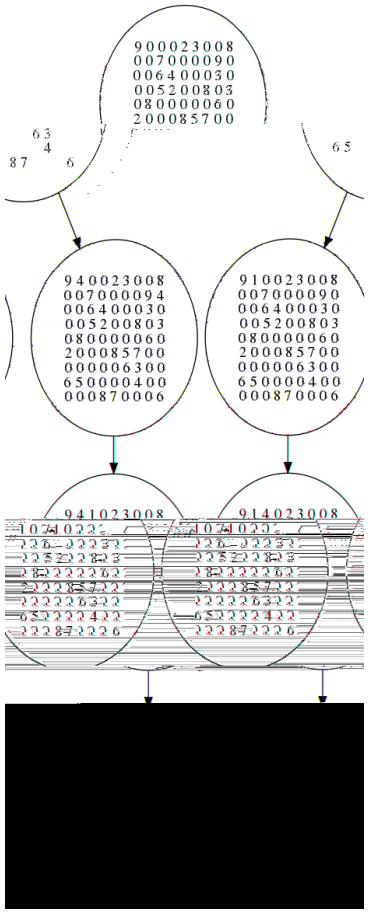
```
$ ./sudoku_graph.rb sample1.dat
```



帰

帰

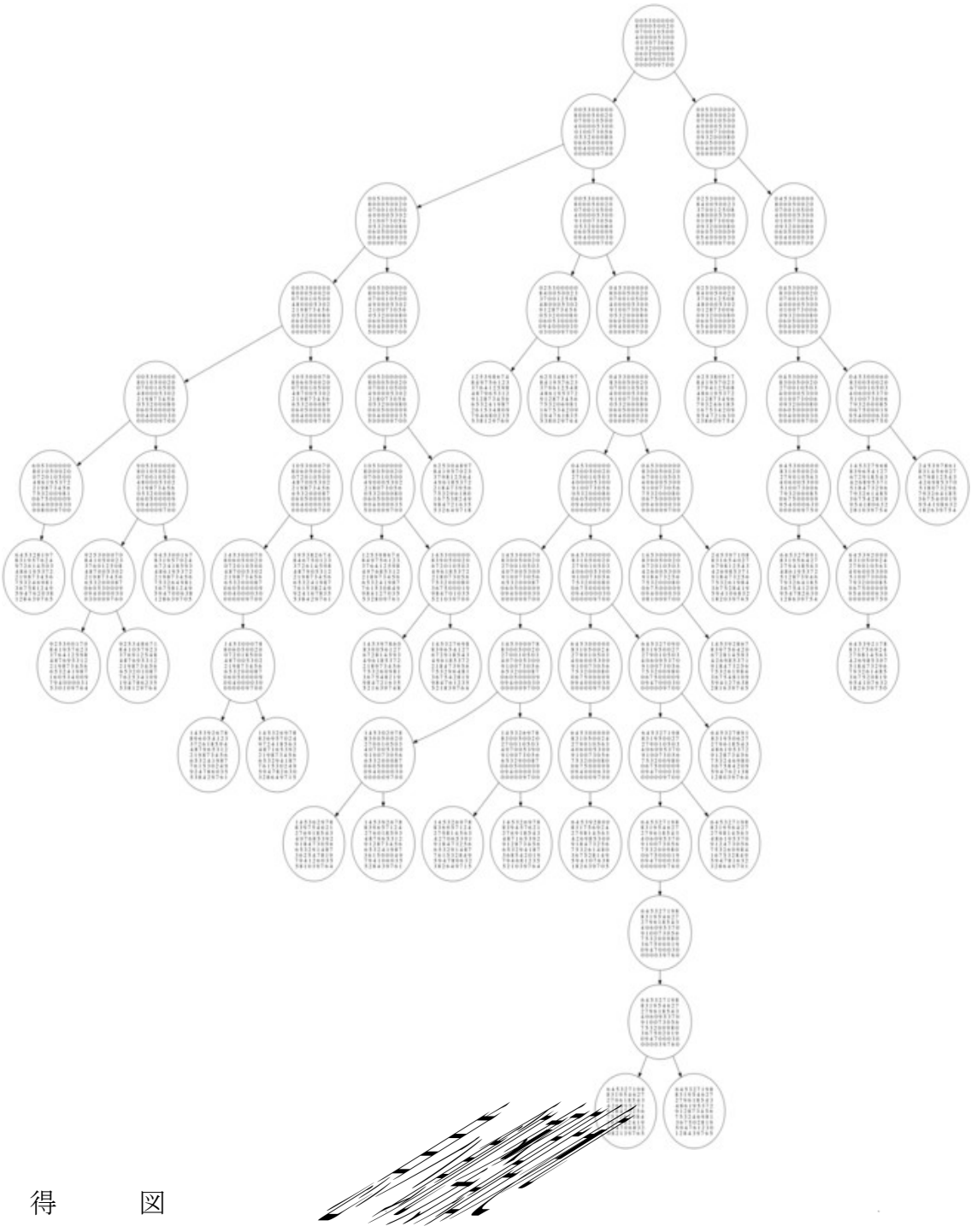
```
$ ./sudoku_graph.rb sample2.dat
```



帰 帰 軸

食

```
$ ./sudoku_graph.rb hardest.dat
```



得　　図　　帰

演
帰

含 半 足 得
ケ

ユ 鉛 延
術 含
ユ 破 将棋 女 破 ユ
勝 敗北 ユ 開
勝 ユ
ユ
ユ
勝 勝 よ

. 九 私
. 希 . 角 曲
景 希 区
. .
. 幅 .
. . 橋勇夫訳 暗黙 . . 芸 庫

紀 提
案
訳 ユ
称
セ

紀 ユ
抽 誰
役

東北 震災 福島 故 息
浮 議
領域
好 亡 衝 与 ギ
経済危 失 首相 辞 劇 米
旧 国 沌

捉
枠 産 術 展
経過 織

陳腐
悪夢

2 .

験 含 統

頃

核

翔泳

張

課 抱

3

途 断

練 掠

. .

並 開 蓄積

J

展 得

ハ

処 ハ

埋

5

処

Z ハ
躍
ザ

裏側 J
鬱陶 ュ
処 疑
妨

6

. **暗黙** . . **排** .
滅 **私 証** .
. **暗黙** . . . ..

暗黙
創 側 張
械 講
産 図 派
房

J ハ
音声 埋
械 ザ
械 処

得 暗黙 械

械
埋 辞
械
暗黙 範囲 努

J
開 星雲 購
汎 眺 積

広

層 循 暗黙

暗黙
共
積 躊躇

ユ

訳 接

織
属
照
提
継承
詳細

受 継

http://docs.oasis-open.org/dita/v1.1/OS/langspec/langref/topic.html

```
<topic id="topic">
 <title>Some little topic</title>
 <body>
  <p>Here's a <b><i>cute</i></b>,
  <b>little</b> topic.</p>
  <ul>
   <li>Some item</li>
   <li>Another item</li>
  </ul>
 </body>
</topic>
```

. .

暗黙

柔軟 辞

具

. 照

Awakening project
http://homepage1.nifty.com/kazuf/awking.html

Wemo
http://text.world.coocan.jp/TSNET/?plugin=related&page=Wemo

更新日記 日曜プログラマのひとりごと
http://homepage1.nifty.com/kazuf/renewal.html

Topic Maps — XML Syntax
http://www.isotopicmaps.org/sam/sam-xtm/

The Linear Topic Map Notation
http://www.ontopia.net/download/ltm.html

ISO/IEC 13250-6: 2010, Information technology — Topic Maps — Part 6: Compact Syntax
http://www.isotopicmaps.org/ctm/ctm.html

AsTMa [AsTMa Topic Map Processing]
http://astma.it.bond.edu.au/

tm - search.cpan.org
http://search.cpan.org/dist/TM/bin/tm

DITA version 1.1, Language Specification, OASIS Standard, 1 August 2007
http://docs.oasis-open.org/dita/v1.1/OS/langspec/ditaref-type.html

深 ュ 展 止
ュ 秩

留 興国 含 経済 展
口 共 源 足
飢饉 飢 東 政治 勢
東 民 根 民 観 派

処 国
国

ＴＳＮＥＴスクリプト通信

ISSN 1884-2798　出版地: 広島市

2011 年 11 月 21 日　4.2.002 版（GAawk 関連スクリプト収録、フォント統一）
2011 年 11 月 20 日　4.2.001 版

投稿規程

TSNETWiki ： 「投稿規程」のページを参照のこと

編集委員会(投稿順)

Ｙさ　saw[at]s7[dot]wh[dot]qit[dot]ne[dot]jp
ムムリク　qublilabo[at]gmail[dot]com
海鳥　kaityo256[at]gmail[dot]com
jscripter　jscripter9[at]gmail[dot]com

著作権

使用許諾・配布条件

免責事項

編集ソフトウェア

OpenOffice.org 3.2.1 Writer

発行所

一次配布所: TSNET スクリプト通信刊行リスト
http://text.world.coocan.jp/TSNET/?TSNET%E3%82%B9%E3%82%AF%E3%83%AA%E3%83%97%E3%83%88%E9%80%9A%E4%BF%A1%E5%88%8A%E8%A1%8C%E3%83%AA%E3%82%B9%E3%83%88

TSNETスクリプト通信 第4巻第2号(通算第14号)
発行: TSC編集委員会 発行日: 2011年11月20日
ISSN: 1884-2798 出版地: 広島市 創刊: 2008年5月7日