c:

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
#include<assert.h>
#define REP(i,a,b) for(i=a;i<b;i++)
#define rep(i,n) REP(i,0,n)

#define ll long long

#define RAND (rand()/(RAND_MAX+1.0))

#define BIG_INT_SIZE 20
#define BIG_INT_BASE 100000000LL
#define BIG_INT_DIGITS 8
#define BIG_INT_CHAR_SIZE 200

typedef struct big_integer{ll a[BIG_INT_SIZE];}bigInt;

int bigIntSign(bigInt a);
int bigIntToChar(bigInt a,char ret[]);
void printBigInt(bigInt a);
void putBigInt(bigInt a);


bigInt bigIntZero(){
  bigInt a; int i;
```

```
  rep(i,BIG_INT_SIZE) a.a[i]=0;
  return a;
}

bigInt bigIntOne(){
  bigInt a; int i;
  REP(i,1,BIG_INT_SIZE) a.a[i]=0; a.a[0]=1;
  return a;
}

bigInt bigIntOrder(bigInt a){
  int i; ll k;
  REP(i,1,BIG_INT_SIZE) if(a.a[i-1]<0 || a.a[i-1]>=BIG_INT_BASE){
    k=a.a[i-1]/BIG_INT_BASE; a.a[i-1]-=k*BIG_INT_BASE;
    if(a.a[i-1]<0) k--, a.a[i-1]+=BIG_INT_BASE; a.a[i]+=k;
  }
  if(a.a[BIG_INT_SIZE-1]<0){
    rep(i,BIG_INT_SIZE) a.a[i]=-a.a[i]; a=bigIntOrder(a);
    rep(i,BIG_INT_SIZE) a.a[i]=-a.a[i];
  }
  return a;
}

bigInt llToBigInt(ll a){
  bigInt c; int i;
  REP(i,1,BIG_INT_SIZE) c.a[i]=0; c.a[0]=a;
  return bigIntOrder(c);
}
```

```
int bigIntGreaterThan(bigInt a,bigInt b){
  int i;
  for(i=BIG_INT_SIZE-1;i>=0;i--){
    if(a.a[i]>b.a[i]) return 1;
    if(a.a[i]<b.a[i]) return 0;
  }
  return 0;
}

int bigIntIsZero(bigInt a){
  int i; rep(i,BIG_INT_SIZE) if(a.a[i]) return 0; return 1;
}

bigInt bigIntPlus(bigInt a,bigInt b){
  int i; bigInt c;
  rep(i,BIG_INT_SIZE) c.a[i]=a.a[i]+b.a[i];
  return bigIntOrder(c);
}

bigInt bigIntMinus(bigInt a,bigInt b){
  int i; bigInt c;
  rep(i,BIG_INT_SIZE) c.a[i]=a.a[i]-b.a[i];
  return bigIntOrder(c);
}

bigInt bigIntMultipleLL(bigInt a,ll b){
  int i; rep(i,BIG_INT_SIZE) a.a[i]*=b;
  return bigIntOrder(a);
}
```

```
bigInt bigIntPlusSimple(bigInt a,bigInt b){
  int i; bigInt c;
  rep(i,BIG_INT_SIZE) c.a[i]=a.a[i]+b.a[i];
  return c;
}

bigInt bigIntMinusSimple(bigInt a,bigInt b){
  int i; bigInt c;
  rep(i,BIG_INT_SIZE) c.a[i]=a.a[i]-b.a[i];
  return c;
}

bigInt bigIntMultipleLLSimple(bigInt a,ll b){
  int i; rep(i,BIG_INT_SIZE) a.a[i]*=b;
  return a;
}

bigInt bigIntMultiple(bigInt a,bigInt b){
  int i,j,ii,jj; bigInt c;
  for(ii=BIG_INT_SIZE-1;ii;ii--) if(a.a[ii]) break; ii++;
  if(ii==1) return bigIntMultipleLL(b,a.a[0]);
  for(jj=BIG_INT_SIZE-1;jj;jj--) if(b.a[jj]) break; jj++;
  if(jj==1) return bigIntMultipleLL(a,b.a[0]);
  rep(i,BIG_INT_SIZE) c.a[i]=0;
  rep(i,ii)if(a.a[i])for(j=0;j<jj&&i+j+1<BIG_INT_SIZE;j++) c.a[i+j]+=a.a[i]*b.a[j];
  return bigIntOrder(c);
}
```

```
void bigIntDivisionsLL(bigInt a,ll b,bigInt *c,ll *d){
  int i;
  rep(i,BIG_INT_SIZE) c->a[i]=a.a[i];
  for(i=BIG_INT_SIZE-1;i;i--)
    c->a[i-1]+=(c->a[i]%b)*BIG_INT_BASE, c->a[i]/=b;
  *d = c->a[0]%b; c->a[0]/=b;
}

/* c=a/b, d=a%b */
void bigIntDivisions(bigInt a,bigInt b,bigInt *c,bigInt *d){
  int i,j,s,sa,sb; ll ma,mb,mc; bigInt tmp;
  sa=bigIntSign(a); sb=bigIntSign(b);
  if(sa==-1) a=bigIntMultipleLL(a,-1);
  if(sb==-1) b=bigIntMultipleLL(b,-1);

  for(j=BIG_INT_SIZE-1;j;j--) if(b.a[j]) break;
  if(!j){
    REP(i,1,BIG_INT_SIZE) d->a[i]=0;
    bigIntDivisionsLL(a,b.a[0],c,&(d->a[0]));
  }else{
    for(i=BIG_INT_SIZE-1;i;i--) if(a.a[i]) break;
    s=i-j; if(s<0) s=0;
    rep(i,BIG_INT_SIZE) c->a[i]=0;
    while(s>=0){
      ma=0; mb=BIG_INT_BASE-1;
      while(ma!=mb){
        mc = (ma+mb)/2 + (ma+mb)%2;
        c->a[s]=mc; tmp=bigIntMultiple(*c,b);
        if(bigIntGreaterThan(tmp,a)) mb=mc-1; else ma=mc;
```

```
    }
    c->a[s]=ma; s--;
   }
   tmp = bigIntMultiple(b,*c);
   *d = bigIntMinus(a,tmp);
  }

  if(sa==-1 && sb==-1){
   *d=bigIntMultipleLL(*d,-1);
  } else if(sa==-1 && sb!=-1){
   *c=bigIntMultipleLL(*c,-1);
   *d=bigIntMultipleLL(*d,-1);
  } else if(sa!=-1 && sb==-1){
   *c=bigIntMultipleLL(*c,-1);
  }
}

bigInt bigIntDivision(bigInt a,bigInt b){
  bigInt c,d;
  bigIntDivisions(a,b,&c,&d);
  return c;
}

bigInt bigIntModular(bigInt a,bigInt b){
  bigInt c,d;
  bigIntDivisions(a,b,&c,&d);
  return d;
}
```

```
int bigIntSign(bigInt a){
  int i;
  for(i=BIG_INT_SIZE-1;i>=0;i--) if(a.a[i]){
    if(a.a[i]<0) return -1; else return 1;
  }
  return 0;
}

bigInt bigIntAbs(bigInt a){
  if(bigIntSign(a)==-1) return bigIntMultipleLL(a,-1LL); return a;
}

bigInt bigIntGCD(bigInt a,bigInt b){
  if(bigIntSign(a)==-1) a=bigIntMultipleLL(a,-1);
  if(bigIntSign(b)==-1) b=bigIntMultipleLL(b,-1);
  if(bigIntIsZero(a)) return b;
  return bigIntGCD(bigIntModular(b,a),a);
}

int bigIntToChar(bigInt a,char ret[]){
  int i,j,s=0,len=0; char ct[BIG_INT_CHAR_SIZE]; ll lt;
  if(bigIntSign(a)==-1){
    ret[0]='-'; len=bigIntToChar(bigIntMultipleLL(a,-1LL),ret+1); return len+1;
  }
  rep(i,BIG_INT_SIZE){
    lt=a.a[i]; rep(j,BIG_INT_DIGITS) ct[s++]=lt%10, lt/=10;
  }
  j=0;
  while(s--){
```

```
    if(ct[s]) j=1;
    if(j) ret[len++]=ct[s]+'0';
  }
  if(!len) ret[len++]='0';
  ret[len]='\0'; return len;
}

void printBigInt(bigInt a){
  int i,k; char tmp[BIG_INT_CHAR_SIZE];
  k=bigIntToChar(a,tmp); rep(i,k) putchar(tmp[i]);
}

void putBigInt(bigInt a){
  char tmp[BIG_INT_CHAR_SIZE];
  bigIntToChar(a,tmp); puts(tmp);
}

bigInt bigIntDivisionLL(bigInt a,ll b){
  bigInt res; ll tmp;
  bigIntDivisionsLL(a, b, &res, &tmp);
  return res;
}

bigInt bigIntSqrt(bigInt a){
  bigInt c1=bigIntZero(),c2=a,c,mul;

  while( bigIntGreaterThan(c2,c1) ){
    c = bigIntDivisionLL( bigIntPlus(bigIntPlus(c1,c2),bigIntOne()), 2);
    mul = bigIntMultiple(c,c);
```

```
   if( bigIntGreaterThan(mul,a) ) c2=bigIntMinus(c,bigIntOne()); else c1=c;
  }

  return c1;
}

bigInt bigIntCubicRoot(bigInt a){
  bigInt c1=bigIntZero(),c2=a,c,mul;

  while( bigIntGreaterThan(c2,c1) ){
    c = bigIntDivisionLL( bigIntPlus(bigIntPlus(c1,c2),bigIntOne()), 2);
    mul = bigIntMultiple(c,bigIntMultiple(c,c));
    if( bigIntGreaterThan(mul,a) ) c2=bigIntMinus(c,bigIntOne()); else c1=c;
  }

  return c1;
}



bigInt in;
int resA[510], resB[510], res_size;

int nowA[710], nowB[710], now_size;
bigInt arr[710];

void solve1(void){
  int i,j;
  arr[0] = bigIntOne();
  now_size = 1;
```

```
  for(i=1;;i++){
    if(bigIntIsZero( bigIntMinus(in,arr[i-1]) )) break;
    for(j=i-1;j>=0;j--){
      arr[i] = bigIntPlus(arr[i-1], arr[j]);
      if(!bigIntGreaterThan(arr[i],in)) break;
    }
    nowA[now_size] = i-1; nowB[now_size] = j;
    now_size++; if(now_size >= res_size) return;
  }

  if(now_size < res_size){
    res_size = now_size;
    rep(i,res_size) resA[i] = nowA[i], resB[i] = nowB[i];
  }
}

void solve_rnd1(double p){
  int i,j,st,dm=-1;
  arr[0] = bigIntOne();
  now_size = 1;
  for(i=1;;i++){
    if(bigIntIsZero( bigIntMinus(in,arr[i-1]) )) break;
    st = i-1;
    if(dm == -1 && RAND < p) st = rand()%i;
    if(dm>=0 && dm < st) st = dm-1;
    for(j=st;j>=0;j--){
      arr[i] = bigIntPlus(arr[i-1], arr[j]);
      if(!bigIntGreaterThan(arr[i],in)) break;
      dm = j;
```

```
    }
    nowA[now_size] = i-1; nowB[now_size] = j;
    now_size++;
    if(now_size >= res_size) return;
   }

   if(now_size < res_size){
    res_size = now_size;
    rep(i,res_size) resA[i] = nowA[i], resB[i] = nowB[i];
   }
 }

 void solve_base(void){
   int i,j,k,base,dig;
   int mi, mj, mm, l, tmpp;
   char arr[400]; int arrs;
   int use[2105], cnv[2105], cs;
   bigInt tmp;

   arrs = 0;
   tmp = in;
   while(!bigIntIsZero(tmp)){
    arr[arrs++] = tmp.a[0] % 2;
    tmp = bigIntDivisionLL(tmp, 2);
   }

   for(base=2, dig=1; base<=2048; base*=2, dig++){
    cs = 0;
    now_size = 0;
```

```
rep(k,base+1) use[k] = 0;

for(i=arrs-1;i>=0;i--){
 if(arr[i]==0){
  continue;
 }
 k = 0;
 rep(j,dig){
  if(i-j < 0) break;
  k *= 2;
  if(arr[i-j]) k+=1;
 }
 use[k] = 1;
 i -= j-1;
}

nowA[now_size] = 0; nowB[now_size++] = 0;
cnv[cs++] = 1;

REP(k,2,base) if(use[k]){
 for(;;){
  mm = -1;
  rep(i,cs) REP(j,i,cs){
   if(cnv[i] + cnv[j] > k) break;
   if(mm < cnv[i] + cnv[j]) mm = cnv[i] + cnv[j], mi = i, mj = j;
  }
  if(mi != k){
   rep(i,cs) REP(j,i,cs) REP(l,j,cs) if(cnv[i]+cnv[j]+cnv[l]==k && cnv[j]+cnv[l] > cnv[cs-1]){
    mi = j; mj = l; mm = cnv[j]+cnv[l];
```

```
    }
   }
   nowA[now_size] = mi; nowB[now_size] = mj; now_size++;
   cnv[cs++] = mm;
   if(cnv[cs-1]==k) break;
  }
 }

for(i=arrs-1;i>=0;i--){
  if(arr[i]==0){
   nowA[now_size] = tmpp; nowB[now_size] = tmpp; tmpp = now_size; now_size++;
   continue;
  }
  k = 0;
  rep(j,dig){
   if(i-j < 0) break;
   k *= 2;
   if(arr[i-j]) k+=1;
  }
  rep(mm,cs) if(cnv[mm] == k) break;

  if(i==arrs-1){
   tmpp = mm;
  } else {
   rep(mi,j){
    nowA[now_size] = tmpp; nowB[now_size] = tmpp; tmpp = now_size; now_size++;
   }
   nowA[now_size] = tmpp; nowB[now_size] = mm; tmpp = now_size; now_size++;
  }
```

```
      if(now_size >= res_size) break;

      i -= j-1;
    }

    if(res_size > now_size){
      res_size = now_size;
      rep(i,res_size) resA[i] = nowA[i], resB[i] = nowB[i];
    }
  }
}

int main(){
  int i,j,k,l,m,n;
  char buf[200];

  srand(time(NULL));

  assert( scanf("%s",buf)==1 );
  n = strlen(buf);
  assert(1<=n && n<=100 && buf[0]!='0');
  rep(i,n) assert('0'<=buf[i] && buf[i]<='9');

  in = bigIntZero();
  rep(i,n) in = bigIntPlus( bigIntMultipleLL(in, 10), llToBigInt(buf[i]-'0') );
  res_size = 501;

  solve1();
  rep(i,20) solve_rnd1(RAND*0.15);
```

```
  solve_base();

  printf("%d\n",res_size-1);
  REP(i,1,res_size) printf("%d %d\n", resA[i], resB[i]);

  return 0;
}
```

---

python:

```
n = int(input())
if (n == 1):
    print(0)
    exit(0)
if (n == 2):
    print(1)
    print(0, 0)
    exit(0)
n1 = n
s = []
while (n1 > 0):
    s.append(n1 % 4);
    n1 //= 4
answer = [(0, 0), (0, 1)]
last = s[-1] - 1
for i in range(len(s) - 2, -1, -1):
    if (last != 0):
```

```
        answer.append((last, last))
        last = len(answer)
    else:
        last = 1
    answer.append((last, last))
    last = len(answer)
    if (s[i] != 0):
        answer.append((last, s[i] - 1))
        last = len(answer)
assert(len(answer) < 500)
print(len(answer))
a = [1]
for i in range(len(answer)):
    print(answer[i][0], answer[i][1])
    a.append(a[answer[i][0]] + a[answer[i][1]])
assert(a[-1] == n)
```