**Контрольная работа (ejudge) от 28.10.2023**

**Задача 1**

*Префиксной* записью выражения называется запись, в которой сначала идет символ операции, затем идет первое подвыражение, затем второе подвыражение.

Например, выражение A+B в префиксной записи выглядит как +AB. Выражение (A+B)\*(C+D) в префиксной записи выглядит как \*+AB+CD. Выражение A-B-C-D в префиксной записи выглядит как ---ABCD. Выражение A-(B-(C-D)) в префиксной записи выглядит как -A-B-CD.

Напишите программу, которая будет вычислять выражение ---ADC\*BA. Значения A, B, C, D задаются на стандартном потоке ввода по одному значению на строке. Значения A, B, C, D - целые. Гарантируется, что промежуточные и окончательный результаты вычисления выражения представимы 64-битным знаковым целым типом. Результат выведите на стандартный поток вывода.

Пример входных данных:

0

0

0

0

Результат работы:

0

Решение:

#include <stdio.h>  
#include <stdlib.h>  
#include <unistd.h>  
#include <limits.h>  
  
int main(void){//(a\*b)\*c\*d+b // a-d-c-b\*a  
    long long A, B, C, D;  
    scanf("%lld%lld%lld%lld", &A, &B, &C, &D);  
    printf("%lld\n", A\*B\*C\*D+B /\* a-d-c-b\*a \*/);  
    return 0;  
}

**Задача 2**

На стандартном потоке ввода подаётся целое число N (0 <= N <= 1000) и целое число M (0 <= M <= 1000). Далее следуют N 64-битных беззнаковых целых числа X[i] в восьмеричной записи, после которых M 64-битных беззнаковых целых числа Y[j] в восьмеричной записи. Далее следует 64-битное беззнаковое целое число S в восьмеричной записи.

На стандартный поток вывода напечатайте в десятичном виде два индекса i и j такие, что сумма чисел X[i] и Y[j], рассматриваемых как 128-битные числа, равна S. Если таких пар индексов несколько, выберите лексикографически максимальную. Если такой пары не существует, выведите два числа -1.

### Examples

#### Input

2 2 1 2 3 4 5

#### Output

1 0

Решение:

#include <stdio.h>

unsigned long long X[1000] = {0};

unsigned long long Y[1000] = {0};

int main(int argc, char \*\* argv){

int N, M, ires = -1, jres = -1, i, j;

unsigned long long S, tmax, sum;

if (scanf ("%d%d", &N, &M) !=2 || N == 0 || M == 0)

return 0;

for (i = 0; i < N; ++i){

if (!scanf ("%llo", &X[i]))

return 0;

}

for (j = 0; j < M; ++j){

if (!scanf ("%llo", &Y[j]))

return 0;

}

if (!scanf ("%llo", &S))

return 0;

for (i=0; i < N; ++i)

for(j = 0; j< M; j++){

tmax = X[i]<Y[j] ? Y[j] : X[i];

sum = X[i] + Y[j];

if (sum >= tmax && sum == S){

ires = i;

jres = j;

}

}

printf ("%d %d\n", ires, jres);

return 0;

}

**Задача 3**

Напишите функцию

void back\_max(char \*str);

Функция принимает строку str, в которой записаны 32-битные знаковые целые числа в десятичной записи, разделенные запятыми.

Функция должна преобразовать строку так, чтобы последнее по порядку максимальное число в этой строке было переставлено в конец строки. Например, если на входе получена строка "5,4,5,2,1,3", после преобразования строка должна быть равна "5,4,2,1,3,5".

Строка состоит только из десятичных цифр, знаков "минус" и запятая. Незначащие нули в начале чисел отсутствуют

Решение:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

void back\_max(char \*str){

int maxi = 0, i = 0,n, max;

if(sscanf(str, "%d", &max)!=1)

return;

while(i< strlen(str)){

if(str[i]!=','){

i++;

continue;

}

i++;

if (sscanf(str+i,"%d", &n) != 1)

return;

if(n >= max){

max = n;

maxi = i;

}

}

char t[20];

int lenmax = sprintf(t,"%d", max);

if (maxi+ lenmax == strlen(str))

return;

for(i = maxi; i < strlen(str)-lenmax-1; i++){

str[i] = str[i+lenmax+1];

}

str[i] = ',';

strcpy(str+i+1, t);

}

int main()

{

char \* str = malloc(1024);

// strcpy(str, "5,4,5,2,1,3,9");

strcpy(str, "5,3,4");

back\_max(str);

printf("%s\n", str);

strcpy(str, "5");

back\_max(str);

printf("%s\n", str);

free(str);

return 0;

}

**Задача 4**

Программе в аргументе командной строки передаётся имя файла. Файл является бинарным файлом и рассматривается как двоичная запись числа произвольной длины в представлении Little Endian (то есть младшие байты числа идут в файле первыми).

Например, файл с байтовым содержимым 0x12 0x34 0x56 0x78 рассматривается как 32-битное беззнаковое число 0x78563412.

Найдите в записи числа все самые длинные последовательности из единичных битов, идущих подряд, и замените каждый бит в них на ноль.

Решение:

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

size\_t bitNum = 0;

int fd;

unsigned char c;

int get\_next\_bit()

{

if (bitNum%8 == 0) {

if (read(fd,&c,sizeof c) != sizeof c)

return EOF;

}

bitNum++;

int retval = c & 1;

c >>= 1;

return retval;

}

void zero(size\_t startBit, size\_t len)

{

//printf("Clearing %ld bits from %ld\n", len, startBit);

size\_t savedPos = lseek(fd, 0, SEEK\_CUR);

size\_t curPos = startBit/8;

unsigned char cc;

lseek(fd, curPos, SEEK\_SET);

read(fd,&cc, sizeof cc);

//printf("Read:%2x\n",cc);

lseek(fd,curPos, SEEK\_SET);

size\_t finBit = startBit + len;

while (startBit < finBit) {

int byteBit = startBit % 8;

cc &= ~(1 << byteBit);

startBit++;

if (startBit % 8 == 0) {

write(fd, &cc , sizeof cc);

//printf("Write:%2x\n",cc);

read(fd,&cc, sizeof cc);

lseek(fd, - sizeof cc, SEEK\_CUR);

//printf("Read:%2x\n",cc);

}

}

write(fd,&cc, sizeof cc);

//printf("Write:%2x\n",cc);

lseek(fd, savedPos, SEEK\_SET);

}

int main(int argc, char \*\* argv)

{

if (argc < 2)

return 0;

fd = open(argv[1], O\_RDWR);

if (fd<0)

return 0;

size\_t maxLen = 0; int b; int inSeq = 0; size\_t curLen = 0;

while ((b = get\_next\_bit()) != EOF) {

if (b) {

curLen++;

inSeq = 1;

} else if (inSeq) {

inSeq = 0;

if (curLen > maxLen)

maxLen = curLen;

curLen = 0;

}

}

if (curLen > maxLen)

maxLen = curLen;

//printf("found seq of len %ld\n", maxLen);

bitNum = 0;

curLen = 0;

inSeq = 0;

size\_t seqStart = 0;

lseek(fd,0, SEEK\_SET);

while ((b = get\_next\_bit()) != EOF) {

if (b) {

if (!inSeq)

seqStart = bitNum - 1;

inSeq = 1;

curLen++;

} else if (inSeq) {

inSeq = 0;

if (curLen == maxLen)

zero(seqStart, maxLen);

curLen = 0;

}

}

if (inSeq && curLen == maxLen)

zero(seqStart, maxLen);

close(fd);

return 0;

}