Шеллом Записать в мэйн ˅

//mass = ShellSort(mass, size);

void shell(int \*arr, int n)

{

for (int step = n / 2; step; step /= 2)

for (int i = step; i < n; ++i)

for (int j = i - step; j >= 0 && arr[j] > arr[j+step]; j -= step){ temp = arr[j];

arr[j] = arr[j+1];

arr[j+1] = temp;} return arr;}

Хоар

void sort(int\* nums, int begin, int end)

{int l = begin, r = end;

int v = nums[l+(r-l)/2];

while(l <= r)

{while(nums[l] < v) l++;

while(nums[r] > v) r--;

if(l <= r)

{int tmp = nums[l];

nums[l] = nums[r];

nums[r] = tmp;

l++, r--;}}

if(begin < r)sort(nums, begin, r);

if(l < end)sort(nums, l, end);}

//ф-ция для обмена значений ячеек

void swapEl(int \*arr, int i)

{ int buff;

buff = arr[i];

arr[i] = arr[i - 1];

arr[i - 1] = buff;}

//ф-ция "шейкер"-сортировки

void myShakerSort(int \*arr, int size)

{ int leftMark = 1;

int rightMark = size - 1;

while (leftMark <= rightMark)

{ for (int i = rightMark; i >= leftMark; i--){

if (arr[i - 1] > arr[i]) swapEl(arr, i);

leftMark++;}

for (int i = leftMark; i <= rightMark; i++)

{ if (arr[i - 1] > arr[i]) swapEl(arr, i);

rightMark--;}}}

//Выбором/отбором (мин элемент на др.)

void otbor(int arr[], int n)

{for (int i = 0; i < n - 1; ++i)

for (int j = i + 1; j < n; ++j)

if (arr[i] > arr[j]){ temp = arr[i];

arr[i] = arr[j]; [j] = temp;}}

// Вставками (в гугле)

void vstavka(int arr[], int n){

for (int i = 1; i < n; ++i)

for (int j = i; j && arr[j] < arr[j-1]; --j){

int temp = arr[j];

arr[j] = arr[j-1]; arr[j-1] = temp;}}

// Пузырёк (если прям совсем куку)

void bubbleSort(int \*num, int size)

{for (int i = 0; i < size - 1; i++){

for (int j = (size - 1); j > i; j--){

if (num[j - 1] > num[j]){

int temp = num[j - 1];

num[j - 1] = num[j];

num[j] = temp;}}}}

void perevodsyss(double num, int ss, int toch)

{int znak;

if (num < 0){znak = -1;

num \*= -1;} int i\_num = (int) num;

num -= i\_num; char str[2048];

int i = 0, j; do{j = i\_num % ss;

str[i++] = j < 10 ? j + '0' : j - 10 + 'a';

i\_num /= ss;}while(i\_num > 0);

if (znak == -1){str[i++] = '-';}

strnrev(str, i); str[i++] = '.';

while(num != 0, toch){num \*= ss;

j = (int)num; num -= j;

str[i++] = j < 10 ? j + '0' : j - 10 + 'a';

toch--;} str[i++] = '\0'; puts(str);}

void strnrev(char \*str, int n){char \*tmp = str + n - 1;

while (str < tmp) { char c = \*str;

\*str++ = \*tmp; \*tmp-- = c;}}

// Перевод в другую сс рекурсивно

char\* f(int n, int k, int p, char\*s);

main()

{int n, p, k, n1;

char \*s;

printf("Enter n, p: ");

scanf("%d%d", &n, &p);

k=0; n1=n; while (n>0){n/=p; ++k;}

s=(char\*) calloc(k+1, sizeof(char));

s=f(n1, k-1, p, s);

printf("Result: \n");puts(s);

free(s); getch();return 0;}

char\* f(int n, int k, int p, char\*s)

{ int ost; if (n>0){ ost=n%p;

if (ost<10){s[k]=ost+'0';}

else {s[k]=ost-10+'A';}

f(n/p, k-1, p, s); }

return s;}

//Перемножение матриц

int main(void)

{int a[3][4], a1[4][2], a2[3][2], i = 0, j = 0, k = 0;

/\*"функция сканирования для a, a1"\*/

do{j = 0;

do{a2[i][j] = 0;

k = 0;

do{a2[i][j] += a[i][k] \* a1[k][j];

k++;}while (k<4);

j++;}while(j<2);

i++;}while(i<3);

//ф-ция вывода a2

return 0;}

//Седловые точки матрицы

void sedlo(int \*\*a, int m, int n)

{int i, j, k, is\_sedlo;

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

for(j = 0; j < n; j++){is\_sedlo = 1;

for(k = 0; k < m; k++)

{ if(a[i][j] < a[k][j] && k!=i) is\_sedlo = 0; }

for(k = 0; k < n; k++)

{ if(a[i][j] > a[i][k] && k!=j) is\_sedlo = 0;}

if(is\_sedlo){ printf("%d ", a[i][j]);}}}

return;}

//char\* в short int\* ( for (int i = 0; i < length; i++) //{printf("%hd ", array[i]); })

int strtoshort(char \*str, short \*arr)

{int i = 0, znak = 1, razr = 0, j = 0;

for(i = 0; str[i]; i++){

if (str[i] == '-') znak = -1;

if(str[i]>= '0' && str[i] <= '9')

razr = 10 \* razr + (str[i] - '0');

if(str[i] == ' ' || str[i] == '\0')

{arr[j] = (znak \* razr);

razr = 0;

znak = 1;

j++;}}

return arr[j] = (znak \* razr);}

//Фибоначчи рекурсией

for(int i = 0; i < n; i++){ printf("%d, ", fibon(i));}

int fibon(int n)

{ return (n == 0 || n == 1)? 1:fibon(n-1)+fibon(n-2);}

//Високосные года через cmd(atoi написать самому)

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

{int sum = 0;

for (int i = 1; i < argc; i++){

if (atoi(argv[i])%4 == 0)

sum += chartoint(argv[i]);}

printf("Sum is: %d\n", sum);}

//min слово в массиве строк (max по аналогии)

void minslovo(char \*\*str, int nstr, int sstr)

{int i, j = 0, k = 0, dl = 0, mdl = 0, l, imin, msym; // msym - символ начала минимального слова

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

while (str[i] != '\0'){

while(str[i][j] == ' ')

j++;k = j; i = imin;

while (str[i][j] != ' ' && str[i][j] != '\0'){

j++; dl++;}

if (mdl > dl){mdl = dl; msym = k;

imin = i;}dl = 0;} i = 0;}

while (str[imin][msym] != ' ' && str[imin][msym] !='\0')

{printf("%c", str[imin][msym]);

msym++;}}

//Нод рекурсией

int nod(int a, int b)

{int c; if (b > a) c = nod (b, a);

else if (b<=0) c = a;

else{ c = nod (b, a%b); printf("%d", nod(b, a%b));}

return c;}

//Переворот без массива

void f(){static short int pos = 0;

long int num;

if (pos != 0 && getchar() == '\n')

{pos--;return;} scanf("%ld", &num);

pos++; f();

printf("%hi) %li\n", pos+1, num);

pos--;}

// Ф-ция переворота для строки

void revers(char \*str, int ns, int ks)

{ for( ; ns < ks; ns++, ks--)

{ char tmp = str[ns];

str[ns] = str[ks];

str[ks] = tmp;}}

// Ф-ция удаления слова

void sldel(char \*str, int ns, int ks)

{if((str[ns] = str[ks]) != 0)

{ns++; ks++;

puts(str);

return sldel(str, ns, ks);}}

// Транспонирование побоч (остальное по аналогии)

void pobochdowhile(int \*\*mx, int size)

{int i = 0, j = 0;

do{do{ tmp = mx[i][j];

mx[i][j] = mx[size - j - 1][size - i - 1];

mx[size - j - 1][size - i - 1] = tmp; j++; i++

}while(j < size - i - 1) }while(i < size - 1);}

void recour (int \*\*matrix, int size)

{static int i = 0;static int j = 0; int tmp = matrix[i][j];

matrix[i][j] = matrix[size - j - 1][size - i - 1];

matrix[size - j - 1][size - i - 1] = tmp;

if (size - 1 > i++)return j = 0, recour (matrix, size);

if (j < size - i - 1)return recour (matrix, size);

return;}

// Транспонирование главной

void pobochdowhile(int \*\*mx, int size)

{int i = 1, j = 0;

do{do{ tmp = mx[i][j];

mx[i][j] = mx[j][i];

mx[j][i] = tmp; j++; i++;}while(j < i) while(i < size);}

void recur(int \*\*matrix, int size)

{static int i = 1; static int j = 0;

int tmp = matrix[i][j];

matrix[i][j] = matrix[j][i];matrix[j][i] = tmp;

if (++j < i)return recur(matrix, size);

if (++i < size)return j = 0, recur(matrix, size);

return;}

//Сумма интов и даблов через фспчп

void \*\*var\_fun(int n, ...)

{int i, j;

void \*\*sum = (void \*\*)calloc(2, sizeof(void\*));

\*((int\*\*)sum+0) = (int\*)calloc(1, sizeof(int\*));

\*((double\*\*)sum+1) = (double\*)calloc(1, sizeof(double\*));

char c; va\_list p; va\_start(p, n);

while(n--) {c = \*(char \*)p; i = \*(int \*)p;

switch(c) {case 'f': for(j = 0; j < i; j++)

{\*\*((double\*\*)sum+1) += va\_arg(p, double);}

break;

case 'i': for(j = 0; j < i; j++)

{\*\*((int\*\*)sum+0) += va\_arg(p, int);}

break;}} va\_end(p);

printf("%d %f", \*\*((int\*\*)sum), \*\*((double\*\*)sum+1));

return sum;}

// Массив по возрастанию и убыванию в один

int main()

{int mas1[m], mas2[n], mas3[m+n], i1, i2, i3;

printf("Vozr");

//Ввод mas1 и mas2 (m и n через define)

i1 = 0; i2 = m-1; i3 = 0;

while(i1<n && i2>=0)

{while(mas1[i1]<=mas2[i2] && i1<n)

mas3[i3++]=mas1[i1++];

while(mas2[i2]<mas1[i1] && i2>=0)

mas3[i3++]=mas2[i2--];}

while(i1<n) mas3[i3++]=mas1[i1++];

while(i2>=0) mas3[i3++]=mas2[i2--];

//вывод mas3;return 0;}

//Удаление первого и таких же слов cmd

#include <stdbool.h>

int main (int argc, char \*argv[]) {bool repetion = false;

int fs, ss, i=0, fds=0, sds=0;fs=0;

while (argv[1][i]!=' '&&argv[1][0]!='\0')i++;

ss=i+1;

while (argv[1][i]){i++;fds=i;

repetion = checking(argv[1],fds);

while(argv[1][i]!=' '&&argv[1][i]!='\0')i++;

if(argv[1][i]=='\0')sds=i;else sds=i+1;

if(repetion==true){

while((argv[1][fds]=argv[1][sds])!=0){

fds++; sds++;}}}

while((argv[1][fs]=argv[1][ss])!=0){fs++;ss++;}

printf("%s \n", argv[1]);}

bool checking (char\* string, int simbol){int i=0;

while(string[i] != ' '){

if(string[i]!=string[simbol]){return false;}

i++;simbol++;}return true;}

//Транспонирование главной/побочной через указат.

//Ввод

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

for (j = 0; j < size; j++)

scanf("%d", (a + i\*size + j));

//Транспонирование главной

for (i = 0; i < size - 1; i++)

for (j = i + 1; j < size; j++)

{int tmp = \*(a + i \* size + j);

\*(a + i \* size + j) = \*(a + j \* size + i);

\*(a + j \* size + i) = tmp;}

//Транспонирование побочной

for (i = 0; i < size - 1; i++)

for (j = 0; j < size - i - 1; j++)

{int tmp = \*(a + i \* size + j);

\*(a + i \* size + j) = \*(a + (size - j - 1) \* size + (size - i - 1));

\*(a + (size - j - 1) \* size + (size - i - 1)) = tmp;}

//Минимальное слово ФСПЧП

char\* fun(int k, ...)

{int i=0,begin=0,end=0,min\_begin=0,min\_end=0,min=0;

char \*word,\*str; va\_list p; va\_start(p, k);

word = (char\*)malloc(1 \* sizeof(char));

str = va\_arg(p, char\*);int m = 0;

while (str[m] == ' '){m++;} min\_begin = m;

while (str[m] != ' ' && str[m] != '\0')

{m++;}min\_end = m -1;

min = min\_end - min\_begin;

for (int j = 0; j < k; j++){m = 0;

while (str[m]){while (str[m] == ' ')

{m++;}begin = m;

while (str[m] != ' ' && str[m] != '\0')

{m++;}end = m - 1;

if ((end - begin) < min){min = end - begin;

min\_end = end;min\_begin = begin;int n,i1;

word=(char\*)realloc(word, min \* sizeof(char));

for(i1=0, n=min\_begin; n<=min\_end;i1++,n++)

{word[i1]=str[n];}word[n]='\0';}}

str = va\_arg(p,char\*);}va\_end(p); return word;}

int main(){int j,i,k=3;char\*\*text;

text=(char\*\*)malloc(k\*sizeof(char\*));

for (i=0;i<k;i++){text+i)=(char\*)malloc(1\*sizeof(char));}

rewind(stdin);j=0; for (i=0;i<k;i++){

while ((text[i][j++]=getchar())!='\n')

{text[i]=(char\*)realloc(text[i],(j+1) \*sizeof(char));}

text[i][j-1]='\0';j=0;}

printf("%s",fun(3,text[0],text[1],text[2]));}

//Большую строку в меньшую с n (вычислить длинну)

//Выделить память как size\*2

scanf("%d", &n);

if(len2 < len1)strob(str2, str1, n, len2);

else if(len1 <= len2)strob(str1, str2, n, len1);

puts(str1);return 0;}

void strob(char \*str1, char \*str2, int n, int size)

{int i;

for(i = n; i < n + size; i++)

str1[i] = str2[i - n];}

//Меняем первое и последнее слово

revers(str, FirstWordN, FirstWordK);

revers(str, LastWordN, LastWordK);

revers(str, FirstWordK + 1, LastWordN - 1);

revers(str, FirstWordN, LastWordK);

puts(str); return 0;}

void revers(char \*str, int ns, int ks)

{if(ns < ks){char tmp = str[ns];

str[ns] = str[ks];

str[ks] = tmp;ns++;ks--;

return revers(str, ns, ks);}}

//Расчёт по формуле

static int factorial (int n);

float power(float x, int y);

int main()

{float e, x, n = 1;int k;

//Сканф e, x, k

float res = x, lastRes;

do{lastRes = res;

res = lastRes + (power(-1, n) \* factorial(2 \* n + 1) / power(x - 2 \* n,2 \* n));

printf("%.0f\t%f\n", n, res);n++;}while (abs(res - lastRes) > k && res != e);

printf("%lf", res); return 0;}

static int factorial (int n)

{return (n < 2) ? 1 : n \* factorial (n - 1);}

float power(float x,int y){int i;float rez = 1;

for(i = 1; i <= y; ++i){rez \*= x;}

return rez;}

//Сортировка через CMD

#include <stdbool.h>

bool compare(char\* word1, char\* word2)

{int i = 0;while (true)

{if (word1[i] == '\0' && word2[i] != '\0') return false;

else if (word2[i] == '\0' && word1[i] != '\0') return true;if (word1[i] < word2[i]) return false;

if (word2[i] < word1[i]) return true;i++;}}

int main(int n, char \*\*str)

{for (int i = 1; i < n; ++i)

printf("%s\t", str[i]);puts("");

for (int i = 1; i < n - 1; ++i)

for (int j = n - 1; j > i ; --j)

if (compare(str[j-1], str[j])){char \*temp = str[j];

str[j] = str[j-1];str[j-1] = temp;}

for (int i = 1; i < n; ++i) printf("%s ", str[i]);

return 0;}

//Удаление лишних пробелов

void fun(char\* str)

{static int i = 0, i1 = 0, j1 = 0, flag;

if (str[i]){if (j1- i1 > 1 && flag){

if (str[j1]){

str[i1 + 1] = str[j1];

i1++; j1++;}else{str[i1 + 1] = '\0';

flag = 0;}fun(str);}

if (str[i] && str[i] != ' '){ i1 = i;i++;flag = 0;fun(str);}

if (str[i] == ' ') {j1 = i;i++;flag = 1;fun(str);}}}

//Сумма первых 5 нечётных под главной диаг.

int fun(int\*\* mt, int s)

{static int i = 1, j, count = 5, sum;

if (i < s) {if (j < i){if (mt[i][j] % 2 != 0 && count)

{sum += mt[i][j];count--;}j++;fun(mt, s);}

else{i++; j = 0;fun(mt, s);}}

return sum;}

//Нахождение стр., в кот. больше слов, нач. с большой русской буквы fun(argv, argc)

int fun(char\*\* s, int n)

{static char\*\* p;

static int max, med, flag, num, maxnum, word;

if(n){if (!word){flag = 1;p = s;}

if (\*\*p){if (\*\*p == ' '){(\*p)++;flag = 1;word = 0; fun(s, n);}

if (\*\*p && \*\*p != ' ')

{if(\*\*p >= 'A' && \*\*p <= 'Z' && flag == 1){

med++;(\*p)++;flag = 0;word = 1; fun(s, n);}

else{(\*p)++;flag = 0;word = 1; fun(s, n);}}}

if (med >= max){max = med;maxnum = num;}

num++;fun(++s, --n);}return maxnum;}

//Ф-ция перемножения матриц

void func(int\*\* a, int\*\* b, int n, int n1, int m, int m1, int\*\* c)

{int i = 0, j = 0, k, s, i1 = 0, j1 = 0, i2 = 0, j2 = 0;

do{do{do{c[i][j] += a[i1][j1] \* b[i2][j2]; j1++; i2++;

} while (j1 < m);

i2 = 0;j1 = 0;j2++;j++;} while (j < m1);

i2 = j2 = j1 = 0;i1++;i++;j = 0;} while (i < n);

for (i = 0; i < n; i++) {for (j = 0; j < m1; j++)

printf("%d ", c[i][j]); printf("\n");}}

//Реверс побочной диагонали

int\*\* func(int\*\* a, int n) {static int i;

if (i<n/2){int c;c = a[i][n-1-i]; a[i][n-1-i] = a[n-1-i][i];

a[n-1-i][i] = c;i++;return func(a, n);}

return a;}

//Максимальная сумма в массиве массивов ФсПЧП

int function(int arraysNumber, int \*array, int arraySize, ...)

{int \*pointer1 = &arraySize;

int \*\*pointer2 = &array; int maxSum = 0, sum, i;

while (arraysNumber--){sum = 0;

for (i = 0; i < \*pointer1; i++) sum += \*(\*pointer2 + i);

if (maxSum < sum){maxSum = sum;}

pointer2 += 2;pointer1 += 2;}return maxSum;}

//Сортировка строки с доп. Массивом

void sort(char \*in, char \*out)

{char \*max\_word;while((max\_word=find(in)))

{for (; \*max\_word!=' ' && \*max\_word!='\0'; )

{\*(out++) = \*max\_word;

\*(max\_word++) =' ';}\*(out++) =' ';}

\*(out--)='\0';}

//Шейкер строк, начинающихся с чётного числа

void func(int\*\* a,int n)

{int i=0, j, k, left=0, right=n-1, flag=1;

while (i < n){if (a[i][0] % 2==0){

while (left < right && flag){flag = 0;

for (j = left; j < right; j++){if (a[i][j] > a[i][j + 1])

{int c;c = a[i][j]; a[i][j] = a[i][j + 1];

a[i][j + 1] = c; flag = 1;}}right--;

for (j = right; j > left; j--)

{if (a[i][j] < a[i][j - 1]) {int c;c = a[i][j];

a[i][j] = a[i][j - 1]; a[i][j - 1] = c;

flag = 1;}}left++;}

i++;left = 0;right = n - 1;flag = 1;}

else i++;}for (i = 0; i < n; i++){for (j = 0; j < n; j++)

printf("%d ", a[i][j]); printf("\n");}}

//Сортировка i-той строки под побочной диагональю

int main()

{ int\*\* a; int i, j, k,n = 0;scanf\_s("%d", &n);

a = (int\*\*)malloc(n \* sizeof(int\*));

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

a[i] =(int\*) malloc(n \* sizeof(int));

//Ввод матрицы int str,max=0;

scanf\_s("%d",&str); for (i = n - 1; i > n - str; i--)

{ max = a[str][i]; k = i;

for (j = i; j >= n - str; j--)

{if (a[str][j] > max) {max = a[str][j]; k = j;}}

a[str][k] = a[str][i]; a[str][i] = max;}}

//Сортировка гл. диагонали матрицы по убыванию

int main()

{int\*\* a,i,j,k,n=5; a = (int\*\*)malloc(n \* sizeof(int\*\*));

for (i = 0; i < n; i++) a[i] = (int\*)malloc(n \* sizeof(int\*));

//Ввод матрицы

for (i = 1; i < n; i++)

{k = a[i][i];j = i; while (j && k > a[j - 1][j - 1])

{a[j][j] = a[j - 1][j - 1];j--;}a[j][j] = k;}

//Вывод матрицы}

//Наибольшее из чисел в строке

void func(char\* a)

{int i=0,s=0,smax=0,k,n=0;

while (a[i]){s = 0; k = 0;

while (a[i] >= '1' && a[i] <= '9' && a[i])

{s+= (a[i++] - '0');s \*= 10;} s /= 10;

if (s>smax)smax = s; if(a[i])i++;} printf("%d", smax);}

//Сортировка i-той строки матрицы

int main()

{ int\*\* a, n, i, j, m,k;

//Сканф n, m, k

a = (int\*\*)malloc(n \* sizeof(int\*\*));

for (i = 0; i < n; i++) a[i] = (int\*)malloc(m \* sizeof(int\*));

//Ввод матрицы

for (i = 1; i < m; i++)

{j = i;while (a[k][i]< a[k][j - 1] && j-1>=0)

{a[k][j] = a[k][j - 1]; j--;} a[k][j] = a[k][i];}

//Вывод матрицы}

//Типо сортировка строки

bool check(char\* firstWord, char\* lastWord)

{for (int i = 0; ; ++i)

{if ((firstWord[i] != '\0' && firstWord[i] != ' ') && (lastWord[i] == '\0' || lastWord[i] == ' ')) return true;

if ((firstWord[i] == '\0' || firstWord[i] == ' ') && (lastWord[i] != '\0' && lastWord[i] != ' ')) return false;

if (firstWord[i] < lastWord[i]) return false;

if (firstWord[i] > lastWord[i]) return true;}}

void reverse(char\* left, char\* right)

{for ( ; left < right; left++, right--){char temp = \*left;\*left = \*right;\*right = temp;}}

int main()

{while (str[i] == ' ')i++;

char\* firstWordStart = &str[i];

while (str[i] != ' ' && str[i] != '\0')i++;

char\* firstWordEnd = &str[i - 1];

for (int j = i; j < n; ++j)

{while (str[j] == ' ')j++;

char\* lastWordStart = &str[j];

while (str[j] != ' ' && str[j] != '\0')j++;

char\* lastWordEnd = &str[j-1];

if (check(firstWordStart, lastWordStart))

{reverse(firstWordStart, lastWordEnd);

reverse(firstWordStart, firstWordStart + (lastWordEnd - lastWordStart));

reverse(lastWordEnd - (firstWordEnd - firstWordStart), lastWordEnd);

reverse(firstWordStart + (lastWordEnd - lastWordStart) + 1, lastWordEnd - (firstWordEnd - firstWordStart) - 1);

firstWordEnd = firstWordStart + (lastWordEnd - lastWordStart);}}}puts(str);return 0;}