

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Дисциплина: «Архитектура вычислительных систем»

Домашнее задание

**«Программа для выполнения арифметических операций
над комплексными числами, представленными дробями»**

Вариант 16

**Выполнил: Казанцев Никита Олегович,
студент БПИ191, ФКН ПИ, ВШЭ**

Москва 2020

1. Текст задания

Разработать программу реализации 4-х действий арифметики комплексных чисел, представленных дробями (использовать целые со знаком, обеспечить максимальную простоту результата).

2. Описание расчетных методов

Арифметика комплексных чисел в работе поддерживается, благодаря работе со следующими формулами:

1) Формула сложения комплексных чисел: $\left(\frac{a}{b} + \left(\frac{c}{d}\right) * i\right) + \left(\frac{e}{f} + \left(\frac{g}{h}\right) * i\right) = \frac{a*f+b*e}{b*f} + \frac{c*h+d*g}{d*h} * i$ – метод SUM

2) Формула вычитания комплексных чисел: $\left(\frac{a}{b} + \left(\frac{c}{d}\right) * i\right) - \left(\frac{e}{f} + \left(\frac{g}{h}\right) * i\right) = \frac{a*f-b*e}{b*f} + \frac{c*h-d*g}{d*h} * i$ – метод SUBSTRACT

3) Формула умножения комплексных чисел: $\left(\frac{a}{b} + \left(\frac{c}{d}\right) * i\right) * \left(\frac{e}{f} + \left(\frac{g}{h}\right) * i\right) = \left(\frac{a*f+b*e}{b*f} - \frac{c*h+d*g}{d*h}\right) + \left(\frac{a*h+b*g}{b*h} + \frac{c*f+d*e}{d*f}\right) * i$ метод MULT

4) Формула деления комплексных чисел: $\left(\frac{a}{b} + \left(\frac{c}{d}\right) * i\right) : \left(\frac{e}{f} + \left(\frac{g}{h}\right) * i\right) = \frac{\left(\frac{a}{b} + \left(\frac{c}{d}\right) * i\right) * \left(\frac{e}{f} - \left(\frac{g}{h}\right) * i\right)}{\left(\frac{e^2}{f^2} + \left(\frac{g^2}{h^2}\right)\right)}$ – метод DIVISION

Дроби упрощаются с помощью нахождения НОД (алгоритмом Евклида, метод GCD) числителя и знаменателя и последующего деления на НОД обоих чисел.

Более подробное описание методов см. в комментариях в тексте программы (п. 6).

3. Описание входных данных

Входные данные для корректной работы – 8 целых чисел со знаком, разделенные между собой пробелами (строка формата 'a b c d e f g h', где $a/b + c/d * i$ – первое комплексное число, $e/f + g/h * i$ – второе комплексное число).

Если ввод пользователя не соответствует формату, указанному выше, или среди знаменателей введенных дробей присутствуют нули, то выводится сообщение о некорректных данных. В таком случае используются данные по умолчанию.

Диапазон допустимых значений: результат четырех операций обязан помещаться в double word, то есть принадлежать диапазону $[-2^{31}, 2^{31}-1]$, в противном случае происходит переполнение. Так как вычисления с дробями для комплексных чисел довольно громоздкие, корректный результат для всех 4-х операций гарантируется для коэффициентов, лежащих в диапазоне $[-15, 15]$. При взаимно простых числах вне данного диапазона может произойти переполнение.

Примеры корректных исходных данных: «-1 1 3 4 6 -7 3 4», «-2 3 -2 4 10 10 -5 12», «1 5 4 2 -1 5 -3 4»

```
C:\Users\User>C:\Users\User\Desktop\assembler\homework.EXE 2 1 5 6 1 1 1 4
```

1 Пример корректного ввода

```
C:\Users\User>C:\Users\User\Desktop\assembler\homework.EXE erroeeqorqoeoqroqorq
incorrect input. Default values are used. your input:

m = 2/3 + (4/5)*i
n = 6/7 + (8/9)*i
```

2 Пример некорректного ввода

4. Описание выходных данных

Выходные данные представляют собой вывод в консоль следующих строк:

а) введенные числа (с сокращенными коэффициентами) в виде $x/y + (u/v)*i$

```
m = 2/3 + (4/5)*i
n = 6/7 + (8/9)*i
```

3 Пример вывода введенных чисел

б) результаты всех 4-х операций (с сокращенными коэффициентами и в максимально простом виде)

```
m + n = 32/21+(76/45)*i
m - n = -4/21+(-4/45)*i
m * n = -44/315+(1208/945)*i
m / n = 6363/7565+(462/7565)*i
```

4 Пример вывода 4-х операций

5. Описание ключевых переменных

Исходные введенные числа хранятся в переменных $a - h$ (см. п.3). В переменных $tmp[1-4]$ после вызова каждой из реализуемых арифметических операций хранится итоговая дробь вида $tmp1/tmp2 + (tmp3/tmp4)*i$. Строковые переменные sub_res , sum_res , div_res , $mult_res$ сохраняют соответствующее строковое представление результата каждой из операций.

6. Текст программы

```
format PE console
include 'win32ax.inc'

.code

start: ;точка старта

cinvoke GetCommandLine ; получаем указатель на командную строку в регистре eax
mov [commandLine], eax ; перемещаем указатель в переменную

; разбиваем входную строку на аргументы по шаблону
cinvoke sscanf,[commandLine],'%*s %d %d %d %d %d %d %d %d',a,b,c,d,e,f,g,h

;сравниваем все знаменатели с нулями
;если есть хотя бы один ноль - заменяем данные на дефолтные
;
cmp [b],0
je DEFAULT
cmp [d],0
je DEFAULT
cmp [f],0
je DEFAULT
cmp [h],0
je DEFAULT
jmp CORRECT

DEFAULT:

cinvoke printf, "incorrect input. Default values are used. ",10, 0

mov [a], 2

mov [b],3

mov [c],4

mov [d], 5

mov [e], 6

mov [f], 7

mov [g],8

mov [h], 9

CORRECT:
; переводим знаки из знаменателей в числители
;+ не забываем сократить дроби
```

```

call CHANGE_SIGNS

; вычисляем сумму комплексных чисел
call SUM
call CREATE_RES
cinvoke sprintf,sum_res,res

; вычисляем разность комплексных чисел
call SUBTRACT
call CREATE_RES
cinvoke sprintf,sub_res,res

; вычисляем произведение комплексных чисел
call MULT
call CREATE_RES
cinvoke sprintf,mult_res,res

; вычисляем частное комплексных чисел
call DIVISION
call CREATE_RES
cinvoke sprintf,div_res,res

; записываем результат в итоговую строку вывода
cinvoke sprintf, res_str, form_str,[a],[b],[c],[d],[e],[f],[g],[h],sum_res,\
sub_res,mult_res,div_res

; выводим итоговый результат
cinvoke printf, res_str

cinvoke ExitProcess,0

; сформируем итоговую строку с ответом, проверяя знаменатели на единицы,
; + числители на нули
CREATE_RES:
mov [res],0 ; очищаем вспомогательную переменную

; проверяем равен ли числитель реальной части нулю
cmp [tmp1],0
je FIRST0

; проверяем является ли числитель мнимой части нулем
cmp [tmp3],0
je SECOND0

; проверяем является ли первый знаменатель единицей
cmp [tmp2],1
.-----

```

```

je FIRST1

; проверяем является ли второй знаменатель единицей
cmp [tmp4],1
je SECOND1

; в случае если неверно ничего из выше перечисленного
cinvoke sprintf,res,"%d/%d+(%d/%d)*i",[tmp1],[tmp2],[tmp3],[tmp4]
ret

FIRST0:          ;первый числитель это ноль
cmp [tmp3],0     ; проверка является ли второй числитель нулем...
je RET0
cmp [tmp4],1     ;и второй знаменатель единиц
je _01

cinvoke sprintf,res,"%d/%d)*i",[tmp3],[tmp4]    ;если нет - выводим ответ
ret

SECOND0:         ;второй числитель 0 ( а первый 100% не 0)
cmp [tmp2],1     ;проверяем, является ли первый знаменатель 1
je _10
cinvoke sprintf,res,"%d/%d",[tmp1],[tmp2]      ;если нет - выводим ответ
ret

FIRST1:          ;первый знаменатель 1 (числители не нули)
cmp [tmp4],1     ;проверяем второй знаменатель на 1
je _11
cinvoke sprintf,res,"%d+(%d/%d)*i",[tmp1],[tmp3],[tmp4]
ret

SECOND1:         ;только второй знаменатель - 1
cinvoke sprintf,res,"%d/%d+(%d)*i",[tmp1],[tmp2],[tmp3]
ret

_01:             ;первый числитель 0 и второй знаменатель 1
cinvoke sprintf,res,"%d*i",[tmp3]
ret

_10:             ;первый знаменатель 1 и второй числитель 0
cinvoke sprintf,res,"%d",[tmp1]
ret

_11:             ;оба знаменателя 1
cinvoke sprintf,res,"%d+(%d)*i",[tmp1],[tmp3]
ret

RET0:           ; оба числителя - нули

```

```

cinvoke sprintf,res,"0"
ret

;-----
SIMPLIFY_ANS: ;упрощает ответ, записанный в виде tmp1/tmp2 + (tmp3/tmp4)*i
;через вызов simplify для обеих дробей

;упрощение дроби при реальной части
mov eax,[tmp1]
mov ebx,[tmp2]

call SIMPLIFY
mov [tmp1],eax
mov [tmp2],ebx

;упрощение дроби при мнимой части
mov eax,[tmp3]
mov ebx,[tmp4]

call SIMPLIFY
mov [tmp3],eax
mov [tmp4],ebx

ret

;-----
; делаем дробь простой ( Евклид), числитель в регистре eax, знаменатель - в регистре ebx
SIMPLIFY:
mov [x],eax
mov [y],ebx

cmp eax,0 ;если числитель 0 - упрощать нечего
je RETURN

call ABS ;находим НОД
call GCD

cmp eax,1 ;если нод 1 - упрощать нечего
je RETURN

mov edx,0
mov eax, [y] ;иначе делим знаменатель на нод..
div ebx
mov [y],eax

mov eax, [x]
cdq
idiv ebx ;...и числитель тоже делим
mov [x],eax

```

```

RETURN:
mov ebx, [y]          ;возвращаем итог в регистрах eax и ebx
mov eax, [x]
ret

;-----
; операция вычитания
SUBTRACT:
; вычитание тож самое что и сложение, но с обратным знаком второго числа
neg [g]               ;меняем знак у числителя
neg [e]               ;и у знаменателя

call SUM ; коллим сумму - теперь в tmp1, tmp2, tmp3 и tmp4 наш ответ

neg [e]               ;возвращаем на место знаки
neg [g]

ret

;-----
;операция сложения
SUM:
;a/b+(c/d)*i + e/f + (g/h)*i

mov eax, [a]          ;сложение дроби реальной части
mov ebx, [f]          ;a/b+e/f=(a*f+b*e)/(b*f)
imul ebx
mov [tmp1], eax       ;перемнож а и f и записали в tmp1

mov eax, [e]
mov ebx, [b]
imul ebx
add [tmp1], eax       ;перемнож b и e, прибавили к tmp1 - в tmp1
;числитель реальной части

mov eax, [b]
mov ebx, [f]
imul ebx
mov [tmp2], eax       ;перемножили b и f, записали в tmp2 - теперь там будет храниться знаменатель реальной части
;-----
;делаем также для мнимой части
;c/d+g/h=(c*h+g*d)/(d*h)
mov eax, [c]
mov ebx, [h]
imul ebx
mov [tmp3], eax       ;перемножили c и h, записали в tmp3

mov eax, [g]

```



```

mov ebx, [d]
imul ebx
add [tmp3], eax      ;перемножили g и d, прибавили к tmp3, получили числитель мнимой части

mov eax, [d]
mov ebx, [h]
imul ebx
mov [tmp4], eax      ;перемножили d и h, теперь в tmp4 числитель мнимой части

call SIMPLIFY_ANS    ; упростили ответ
ret

;-----
; Операция умножения
MULT:
; (a/b+c/d*i)*(e/f+g/h*i)
; для начала найдем реальную часть в итоговом числе
; перемножаем реальные части чисел
mov ebx, [a]
mov eax, [e]          ; (a/b)*(e/f)=a*e/(b*f)
imul ebx              ; перемножаем a и e и записываем в x
mov [x], eax

mov ebx, [b]          ; перемножаем b и f
mov eax, [f]
mul ebx

mov ebx, eax           ; теперь в ebx записан знаменатель (b*f)
mov eax, [x]           ; в eax - числитель (a*e)

call SIMPLIFY          ; упрощаем нашу дробь
mov [tmp1], eax
mov [tmp2], ebx
;-----
; теперь перемножим мнимые части
mov ebx, [c]          ; (c/d)*(g/h)=c*g/(d*h)
mov eax, [g]
imul ebx              ; перемножаем c и g и записываем в x
mov [x], eax

mov ebx, [d]          ; перемножаем d и h
mov eax, [h]
mul ebx

mov ebx, eax           ; теперь в ebx знаменатель (d*h)
mov eax, [x]           ; в eax - числитель (c*g)

call SIMPLIFY          ; упрощаем нашу дробь
;-----
; теперь сложим полученные на предыдущих шагах дроби и получим реальную часть
; итогового ответа
; tmp1/tmp2 + x/y = (tmp1*y + tmp2*x)/(tmp2*y)
<

```

```

mov eax, [tmp1]      ;в [tmp1] числитель нашей первой дроби
mov ebx, [y]         ;в y сейчас знаменатель второй упрощенной дроби
imul ebx
mov [tmp1],eax       ;перемножим tmp1 и y, рез записываем в tmp1

mov eax,[x]
mov ebx,[tmp2]       ;перемножим tmp2 и x
imul ebx
sub[tmp1],eax        ;вычтем результат из tmp1 и получаем числитель реальной части

mov eax,[y]          ;перемножим tmp2 и y
imul ebx
mov [tmp2],eax       ;получаем знаменатель реальной части и записываем его в tmp2

;-----
; (a/b+c/d*i)*(e/f+g/h*i)
; также найдем мнимую часть
; перемножаем реальную часть первого числа с мнимой частью второго
mov ebx,[a]          ;a/b*g/h=(a*g)/(b*h)
mov eax,[g]          ;перемножаем a и g, записываем в x
imul ebx
mov [x],eax

mov ebx,[b]          ;перемножаем b и h
mov eax,[h]
mul ebx

mov ebx,eax          ;кладем в ebx знаменатель b*h
mov eax,[x]          ;в eax числитель a*g

call SIMPLIFY        ;упростим дробь
mov [tmp3],eax       ;записываем ее в соответствующие переменные
mov [tmp4],ebx
;-----
; теперь перемножим реальную часть второго числа с мнимой частью первого
mov ebx,[c]          ;(c/d)*(e/f)=(c*e)/(d*f)
mov eax,[e]
imul ebx             ;перемножаем c и e, результат кладем в x
mov [x],eax

mov ebx,[d]          ;перемножаем d и f
mov eax,[f]
mul ebx

mov ebx,eax
mov eax,[x]

call SIMPLIFY        ;упростим дробь
;-----
;теперь сложим дроби, полученные ранее, и получим мнимую часть ответа
;tmp3/tmp4 + x/y = (tmp3*y + tmp4*x)/(tmp4*y)
mov eax, [tmp3]

```

```

mov ebx, [y]
imul ebx
mov [tmp3], eax      ;перемножаем tmp3 и y и кладем результат в tmp3

mov eax, [x]          ;перемножаем tmp4 и x
mov ebx, [tmp4]
imul ebx
add [tmp3], eax       ;прибавляем результат к tmp3 и получаем числитель
                      ;мнимой части

mov eax, [y]
imul ebx              ;перемножаем tmp4 и y и получаем знаменатель мнимой части
mov [tmp4], eax

call SIMPLIFY_ANS

ret

;-----
DIVISION:
;при выполнении деления нужно домножить числитель и знаменатель на сопряженное
;=> снизу останется модуль второго числа
;то есть,  $(a/b + (c/d)*i) / (e/f + (g/h)*i) = (a/b + (c/d)*i) * (e/f - (g/h)*i) / (e^2/f^2 + g^2/h^2)$ 

;для того чтобы получить верхнюю часть выражения, вызовем умножение, поменяем знак
;перед мнимой частью для второго числа
neg [g]
call MULT
neg [g]

;посчитаем то, что внизу:  $(e^2/f^2 + g^2/h^2) = (e^2*h^2 + g^2*f^2) / (f^2*h^2)$ 
mov eax, [e]          ;умножаем e на e
mov ebx, [e]
imul ebx
imul [h]              ;результат умножаем на h^2
imul [h]

mov [x], eax          ; в x  $e^2*h^2$ 

mov eax, [g]          ;умножим g на g
mov ebx, [g]
imul ebx
imul [f]
imul [f]              ;и на f^2
add [x], eax          ;прибавим к x, теперь в x числитель дроби

mov eax, [f]
mov ebx, [f]
mul ebx               ;считаем знаменатель - умножим f^2 на h^2
mul [h]
mul [h]

mov ebx, eax

```

```

mov eax,[x]

call SIMPLIFY      ;упростим итоговую дробь

; разделим верхнюю часть на получившуюся дробь и получим ответ
; (tmp1/tmp2+(tmp3/tmp4)*i)/(x/y)=tmp1*y/(tmp2*x)+(tmp3*y/(tmp4*x))*i

mov eax,[tmp1]
mul [y]           ; домножим числитель реальной части на y
mov [tmp1],eax

mov eax,[tmp2]    ; домножим знаменатель реальной части на x
mul [x]
mov [tmp2],eax

mov eax,[tmp3]    ; и числитель мнимой
mul [y]
mov [tmp3],eax

call ABS
mov ebx,[tmp4]
cmp eax,0
je S

call GCD
mov eax,[tmp3]
cdq
idiv ebx
mov [tmp3],eax
mov eax,[tmp4]
cdq
idiv ebx
mov [tmp4],eax

S:
mov eax,[tmp4]    ; и знаменатель мнимой
mul [x]
mov [tmp4],eax

call SIMPLIFY_ANS ;упрощаем ответ
ret

;-----
;перенесем знаки минус в числителе
CHANGE_SIGNS:

mov eax,[b]       ;знаменатель реальной части 1 числа
mov edx,[a]
call SIGN
mov [a],edx
mov [b],eax

```

```

mov ebx, eax

mov eax, [f]      ;знаменатель реальной части 2 числа
mov edx, [e]
call SIGN
mov [e],edx
mov [f],eax

mov eax, [d]      ;знаменатель мнимой части 1 числа
mov edx, [c]
call SIGN
mov [c],edx
mov [d],eax

mov ebx, eax

mov eax, [h]      ;знаменатель мнимой части 2 числа
mov edx, [g]
call SIGN
mov [g],edx
mov [h],eax

mov eax, [a]
mov ebx, [b]
call SIMPLIFY
mov [a], eax
mov [b],ebx

mov eax, [c]
mov ebx, [d]
call SIMPLIFY
mov [c], eax
mov [d],ebx

mov eax, [e]
mov ebx, [f]
call SIMPLIFY
mov [e], eax
mov [f],ebx

mov eax, [g]
mov ebx, [h]
call SIMPLIFY
mov [g], eax
mov [h],ebx

ret

;-----

;перенесем знак знаменателя в числитель
SIGN:
cmp eax,0         ;сравним число с нулем

```

```

j1 N      ;если меньше - изменим знак и у числителя, и у знаменателя
ja R      ;если больше - все ок
N:
neg edx
neg eax
R:
ret

;сохраним в eax положительное значение (модуль) числа
ABS:
cmp eax, 0
j1 M      ;если число меньше нуля - изменим его знак
ret
M:
neg eax
ret
;-----

;алгоритм поиска НОД двух положительных чисел ( Алгоритм евклида)
GCD:
N1: cmp eax, ebx      ;сравниваем два числа
je N3      ;если равны - возвращаемся
ja N2      ;если первое больше - N2

xchg eax, ebx      ;если первое меньше - меняем местами
N2: sub eax, ebx      ;вычитаем из первого (большого) числа второе
jmp N1      ;переходим в начало цикла

N3: ret
;-----

.data
commandLine dd ? ;указатель на командную строку
a dd 0      ;a-числитель реальной части комплексного числа
b dd 0      ;b - знаменатель
c dd 0      ;c - числитель мнимой части комплексного числа
d dd 0      ;d - знаменатель

e dd 0      ;аналогично для второго числа
f dd 0
g dd 0
h dd 0

tmp1 dd ?      ;после каждой операции ответ записывается в эти переменные
tmp2 dd ?      ;в виде tmp1/tmp2+(tmp3/tmp4)*i
tmp3 dd ?
tmp4 dd ?

x dd ?      ;вспомогательные переменные
y dd ?
u dd 1
v dd 1

```

```

sub_res db 512 dup(?)           ;результаты вычислений для каждой из операций
sum_res db 512 dup(?)
mult_res db 512 dup(?)
div_res db 512 dup(?)

res db 512 dup(?)

res_str db 512 dup(?)           ;строка итогового вывода
form_str db "your input:",10,10,"m = %d/%d + (%d/%d)*i",10,10,\
"n = %d/%d + (%d/%d)*i",10,10, "result:",10,10,\
"m + n = %s",10,10,"m - n = %s", 10, 10, "m * n = %s",10,10,"m / n = %s", 0

data import

library user32,'USER32.DLL',\
msvcrt, 'MSVCRT.DLL',\
kernel32, 'KERNEL32.DLL',\
shell32, 'SHELL32.DLL'

import user32,\
MessageBox, 'MessageBoxA'

import msvcrt, \
sprintf, 'sprintf', scanf, 'scanf', printf, 'printf'

import kernel32, \
ExitProcess, 'ExitProcess' ,\
GetCommandLine, 'GetCommandLineA'

end data

```