

Высшая школа экономики
Факультет компьютерных наук

Пояснительная записка

программа, вычисляющая простые числа в диапазоне до беззнакового
машинного слова по методу "Решето Эратосфена Киренского"

Выполнил: Шакуров Семен Сергеевич, БПИ194 / 2 подгруппа.

Текст задания

Разработать программу, вычисляющую простые числа в диапазоне до беззнакового машинного слова по методу "Решето Эратосфена Киренского"

Применяемые расчетные методы

Для решения данной задачи применялся метод "Решето Эратосфена Киренского"

Для нахождения всех простых чисел не больше заданного числа n , следуя методу Эратосфена, нужно выполнить следующие шаги:

1. Выписать подряд все целые числа от двух до n (2, 3, 4, ..., n).
2. Пусть переменная p изначально равна двум — первому простому числу.
3. Зачеркнуть в списке числа от $2p$ до n считая шагами по p (это будут числа кратные p : $2p, 3p, 4p, \dots$).
4. Найти первое незачёркнутое число в списке, большее чем p , и присвоить значению переменной p это число.
5. Повторять шаги 3 и 4, пока возможно.

Теперь все незачёркнутые числа в списке — это все простые числа от 2 до n .

Область допустимых значений

Входное значение максимального числа должно быть в диапазоне от 3 до 65535

Тесты

```
Input max number (>= 3 & <= 65535): wgnk
Max number is incorrect
```

```
Input max number (>= 3 & <= 65535): -1
Max number is incorrect
```

```
Input max number (>= 3 & <= 65535): 66000
Max number is incorrect
```

```

Input max number (>= 3 & <= 65535): 15
Max number = 15
Primes numbers:
      2      3      5      7     11     13

```

```

Input max number (>= 3 & <= 65535): 3000
Max number = 3000
Primes numbers:
      2      3      5      7     11     13     17     19     23     29     31     37     41     43
      47     53     59     61     67     71     73     79     83     89     97     101    103    107
    109    113    127    131    137    139    149    151    157    163    167    173    179    181
    191    193    197    199    211    223    227    229    233    239    241    251    257    263
    269    271    277    281    283    293    307    311    313    317    331    337    347    349
    353    359    367    373    379    383    389    397    401    409    419    421    431    433
    439    443    449    457    461    463    467    479    487    491    499    503    509    521
    523    541    547    557    563    569    571    577    587    593    599    601    607    613
    617    619    631    641    643    647    653    659    661    673    677    683    691    701
    709    719    727    733    739    743    751    757    761    769    773    787    797    809
    811    821    823    827    829    839    853    857    859    863    877    881    883    887
    907    911    919    929    937    941    947    953    967    971    977    983    991    997
   1009   1013   1019   1021   1031   1033   1039   1049   1051   1061   1063   1069   1087   1091
   1093   1097   1103   1109   1117   1123   1129   1151   1153   1163   1171   1181   1187   1193
   1201   1213   1217   1223   1229   1231   1237   1249   1259   1277   1279   1283   1289   1291
   1297   1301   1303   1307   1319   1321   1327   1361   1367   1373   1381   1399   1409   1423
   1427   1429   1433   1439   1447   1451   1453   1459   1471   1481   1483   1487   1489   1493
   1499   1511   1523   1531   1543   1549   1553   1559   1567   1571   1579   1583   1597   1601
   1607   1609   1613   1619   1621   1627   1637   1657   1663   1667   1669   1693   1697   1699
   1709   1721   1723   1733   1741   1747   1753   1759   1777   1783   1787   1789   1801   1811
   1823   1831   1847   1861   1867   1871   1873   1877   1879   1889   1901   1907   1913   1931
   1933   1949   1951   1973   1979   1987   1993   1997   1999   2003   2011   2017   2027   2029
   2039   2053   2063   2069   2081   2083   2087   2089   2099   2111   2113   2129   2131   2137
   2141   2143   2153   2161   2179   2203   2207   2213   2221   2237   2239   2243   2251   2267
   2269   2273   2281   2287   2293   2297   2309   2311   2333   2339   2341   2347   2351   2357
   2371   2377   2381   2383   2389   2393   2399   2411   2417   2423   2437   2441   2447   2459
   2467   2473   2477   2503   2521   2531   2539   2543   2549   2551   2557   2579   2591   2593
   2609   2617   2621   2633   2647   2657   2659   2663   2671   2677   2683   2687   2689   2693
   2699   2707   2711   2713   2719   2729   2731   2741   2749   2753   2767   2777   2789   2791
   2797   2801   2803   2819   2833   2837   2843   2851   2857   2861   2879   2887   2897   2903
   2909   2917   2927   2939   2953   2957   2963   2969   2971   2999

```

Список используемых источников

https://ru.wikipedia.org/wiki/Решето_Эратосфена

Приложение

Текст программы

format PE console

entry start

include 'win32a.inc'

section '.data' data readable writeable

maxNumberLabel db 'Input max number (>= 3 & <= 65535): ', 0

inputFormat db '%u', 0

maxNumError db 'Max number is incorrect', 0xD, 0xA, 0

maxNumberOutputFormat db 'Max number = %u', 0xD, 0xA, 0

mallocError db 'Can not allocate memory', 0xD, 0xA, 0

primesLabel db 'Primes numbers:', 0xD, 0xA, 0

primeOutput db '%u', 0x9, 0

startPrint db 0x9, 0

CrLf db 0xD, 0xA, 0

maxNumber dd 0

primesPointer dd 0

NULL = 0

section '.code' code readable executable

start:

enter 0, 0

;ввод максимального числа

call inputMaxNumber

cmp edx, 0

jne customExit

mov [maxNumber], eax

;выделяем память для массива флагов

mov eax, [maxNumber]

call allocateFlagsMemory

cmp edx, 0

jne customExit

mov [primesPointer], eax

;отсеять составные числа

mov eax, [primesPointer]

mov ebx, [maxNumber]

call findPrimes

;вывести простые числа

mov eax, [primesPointer]

mov ebx, [maxNumber]

call output

;освободить память от массива флагов

mov eax, [primesPointer]

call freeFlagsMemory

;завершение программы

call Exit

;-----

output:

enter 12, 1

mov [ebp-4], eax

mov [ebp-8], ebx

push primesLabel

call [printf]

add esp, 4

push startPrint

call[printf]

add esp, 4

cld

mov esi, [ebp-4]

mov edx, esi

add edx, [ebp-8]

inc edx

mov [ebp-12], edx

mov ecx, 0

;Цикл для вывода

outputCicle:

lodsb

cmp al, 0

jne print

jmp isFinish

print:

cmp ecx, 1

```
je isFinish
push esi
push ecx
push primeOutput
call [printf]
add esp, 4
```

```
pop ecx
pop esi
mov edx, [ebp-12]
```

isFinish:

```
inc ecx
cmp esi, edx
jb outputCicle
```

```
push crlf
call [printf]
add esp, 4
```

```
leave
ret
```

;-----

;Поиск простых чисел с помощью решета Эратосфена

findPrimes:

```
enter 8, 1

mov [ebp-4], eax
add eax, ebx
inc eax
```

```
mov [ebp-8], eax
```

```
;вычеркиваем составные числа
```

```
cld
```

```
;p=2
```

```
mov edx, 2
```

```
;множитель c = 2
```

```
mov ecx, 2
```

Cicle:

```
;x = c*p
```

```
mov eax, edx
```

```
push edx
```

```
mul ecx
```

```
pop edx
```

```
cmp eax, ebx
```

```
jbe strikeNumber
```

```
jmp increase
```

strikeNumber:

```
mov edi, [ebp-4]
```

```
add edi, eax
```

```
mov byte [edi], 0
```

```
;c++
```

```
inc ecx
```

```
jmp Cicle
```

increase:

```
mov esi, [ebp-4]
```

```
add esi, edx
```


inc esi

mov ecx, edx

inc ecx

checkNum:

mov eax, ecx

mul eax

cmp eax, ebx

ja returnPrimes

lodsb

inc ecx

cmp al, 0

jne newFound

jmp checkNum

newFound:

mov edx, ecx

dec edx

mov ecx, 2

jmp Cicle

returnPrimes:

leave

ret

;-----

;Освобидить память от массива флагов

freeFlagsMemory:

enter 0, 1

push eax

call [free]

add esp, 4

leave

ret

;-----

allocateFlagsMemory:

enter 8, 1

;выделить eax+1 байт

inc eax

mov [ebp-4], eax

push eax

call [malloc]

add esp, 4

;проверка

cmp eax, 0

je fail

mov [ebp-8], eax

;инициализация

mov byte [eax], 0

cld

mov edi, eax

inc edi

```
mov edx, [ebp-4]
```

```
add edx, eax
```

```
mov al, 1
```

```
writeTrue:
```

```
stosb
```

```
cmp edi, edx
```

```
jb writeTrue
```

```
mov eax, [ebp-8]
```

```
jmp success
```

```
fail:
```

```
mov edx, mallocError
```

```
jmp return
```

```
success:
```

```
mov edx, 0
```

```
return:
```

```
leave
```

```
ret
```

```
;-----
```

```
;Ввод максимального числа
```

```
inputMaxNumber:
```

```
enter 4, 1
```

push maxNumberLabel

call [printf]

add esp, 4

mov eax, ebp

sub eax, 4

push eax

push inputFormat

call [scanf]

add esp, 8

mov eax, [ebp - 4]

;проверка

cmp eax, 3

jb numberError

cmp eax, 65535

ja numberError

jmp successInput

;ВЫХОД

numberError:

mov edx, maxNumError

jmp returnInput

successInput:

push eax

push maxNumberOutputFormat

call [printf]

```
add esp, 4
pop eax
mov edx, 0
```

```
returnInput:
    leave
    ret
```

```
;-----
```

```
;выход при ошибке
```

```
customExit:
    push edx
    call [printf]
```

```
;ВЫХОД
```

```
Exit:
    call [getch]
    push NULL
    call [ExitProcess]
```

```
;-----
```

```
section '.idata' import data readable
```

```
library kernel, 'kernel32.dll',\
    msvcrt, 'msvcrt.dll'
```

```
import kernel,\
    ExitProcess, 'ExitProcess'
```

```
import msvcrt,\
    printf, 'printf',\
```

```
getch, '_getch',\  
scanf, 'scanf',\  
malloc, 'malloc',\  
free, 'free'
```