107-№1. Алутис Н.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

3. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

4. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_3. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем P1, описанной в каком-то другом модуле. Если значение P1 при этом выйдет за допустимый диапазон, то обнулить его.

НОМЕР 1:

```
01|ВВЦ 103 107 000
02|ПЕР 102 000 020;     102=S
03|ПЕР 100 000 103;     100=X[i]
04|СЛЦ 003 003 019                  1) Будет 03=ПЕР 100 001 103 ???
                                       Надо 04|СЛЦ 003 003 018
05|ПЕР 101 000 100
06|СЛЦ 101 101 020
07|УСЛ 013 008 013
08|ПЕР 101 000 100
09|МОД 101 101 022
10|СЛЦ 101 101 101                  2) Не надо, после МОД есть омега
11|УСЛ 012 013 013
12|СЛЦ 102 102 018; S+1
13|ПЕР 101 000 100
14|ВЧЦ 021 021 018 ;n := n-1
15|ПБ  000 002 000                  3) 15|ПБ  000 003 000 !
16|ВЫВ 102 001 000
17|СТОП 000 000 000
18|00 000 000 001
19|00 000 001 000
20|00 000 000 000
21|00 000 000 107
22|00 000 000 003
```

ОЦЕНКА=3

НОМЕР 2:

процедура:

```
Matrix proc
    push ebp
    mov  ebp,esp;
    push eax
    push ebx
    push ecx
    mov  ecx,[ebp+12]
    mov  ebx,[ebp+8]

    mov  eax,ecx
    inc  eax; n+1 потому что диагональные элементы
L:  cmp  byte ptr [ebx], '0' ;проверка на числа
    jb   No
    cmp  byte ptr [ebx], '9'
    ja   No
    mov  byte ptr [ebx], 1; номер в ведомости = 1
No:
    add  ebx,eax
    loop L ;цикл
    pop ecx ;возвращаем
    pop ebx
    pop eax
    pop ebp
    ret 8
Matrix endp
```

вызов:

```
;mov  eax, 100
push eax                        1) ПРОСТО push 100
;mov eax, offset MAS
push eax                        2) ПРОСТО push offset MAS
call Matrix
```

ОЦЕНКА=6

ЗАДАЧА 3:

```
First_1 macro X:req
    local K, CY4, CY, S, F4
    if (type X) EQ 0
        .err <Bad Argument in Macros First_1>
    endif;
        K=0
    if (type X NE 4) and (type X NE 4 (2)) \
        and (type X NE 4(1))                        1) 4,2,1
        .err <Bad Argument in Macros First_1>
    endif
    for reg, <al,ah,bl,bh,cl,ch,dl,dh,  \
              ax,bx,cx,dx,si,di,bp,sp, \
              eax,ebx,ecx,edx,esi,edi,ebp,esp>
        ifidni <reg>, <X>
            K=1
            exitm
        endif
    endm
    if K EQ 1 ;регистр
        .err <Bad Argument in Macros First_1>
    else
        push eax
        push ebx
        push ecx
        push edx

        mov cl,1
        mov edx, 0;кол-во единиц

        if type X EQ 4 ;double
            mov eax, X
        else if (type X EQ 2) or (type X EQ 1)
            movzx eax, X
        endif

        CY:
            mov ebx, eax
            and ebx, 1
            cmp ebx, 0;                 1) Не надо, уже есть флаги
            je  S
            inc edx
        S:
            shr eax, cl
            cmp eax, 0
            jne CY4
        ;на edx кол-во единиц
            if type X EQ 4
                mov al, 32
            else if type X EQ 2
```

```
                mov al,16
            else
                mov al, 8
            endif
            sub al, dl
            mov cl, 1
            ;на al число нулей после единиц
            mov ebx, 0
            cmp dl, 0
            je F4
        CY4:
            shl ebx, 1
            inc ebx
            sub dl, cl
            cmp dl, 0
            jne CY4
        F4:
            mov cl, al
            shl edx, cl
            if type X EQ 4
                mov X, edx
            else if type X EQ 2
                mov X, dx
            else
                mov X, dl
            endif
        pop edx
        pop ecx
        pop ebx
        pop eax
    endif
endm
```

ОЦЕНКА=6

ЗАДАЧА 4:

```
include console.inc
;номер в ведомости 1
.data
     extern P1:word
.code
;внешняя переменная типа word

public Del_3

Del_3 proc
    push ebp
    mov  epb, esp
    push eax
    push ebx
    push edx
    mov  ax, P1
    xor  dx, dx                    1) Зачем ??? Это НЕ деление
    mov  bx, 1
    imul bx
    jo   OV                        2) jno F
    jmp  F
```

```
OV:
    mov  ax, 0
F:
    mov P1, ax
    pop edx
    pop ebx
    pop eax
    pop ebp
    ret
Del_3 endp
end
```

ОЦЕНКА=6–

107-№2. Балабин Д.С.

1. Выписать вид внутреннего машинного представления целой переменной X:

```
X  dw  -3050
```

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **dd** и выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит нечётное количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) произведение *нечётных цифр* в этом тексте, которые расположены после первого символа '#'.

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес массива A целых знаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму N•A[1]+(N-1)•A[2]+...+1•A[N]. Процедура *обязана* выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req,Y:Req
```

Оно должно в свой параметр X, может быть только форматов `m8`, `m16`, `m32` или `i32`, записывать число "1" (битов со значением единица) во внутреннем машинном представлении параметра Y. Параметр Y может быть форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, например, для `Y db 10101010b` должно получаться X=4. Макроопределение *должно* настраиваться на типы параметров и выдавать необходимые диагностики о неверных типах своих операндов.

---

1. Выписать вид внутреннего машинного представления целой переменной X:
X dw –3050

3050=0000 1011 1110 1010
=1111 0100 0001 0101 +1
=1111 0100 0001 0110
-3050=0001 0110 1111 0100=16F4

ОЦЕНКА=6

2.

```
include console.inc
.data
   X dd ?
.code
Start:
   inint X
   mov eax,X
   mov bx,10
   xor ecx,ecx
   xor edx,edx
L:
   cmp eax,0              1) X=0 => ОДНА цифра !!!
                             Надо сначала делить, потом проверять на 0
   je  R
   inc ecx
   mov dx,0               2) НЕТ, X div 10 > ax !
   mov bx,10
   div bx
   cwd                    3) НЕТ, беззнаковые
   jmp L
R:
   test ecx,1
   je N
Y:
   outstr 'Да'
   jmp Kon
N:
   outstr 'Нет'
Kon:
                          4) exit !!!
```

```
   end Start
```

ОЦЕНКА=2

3.

```
include console.inc
.data
   ch1 db ?
   ch2 db '#'
   ch3 db '.'
   s dd 0
.code
Start:
   xor dx,dx;
L:
   inchar ch1
   movzx cx, ch1                1) Зачем ???
   cmp   cx, '.'
   je    ED
   cmp   cx,'#'
   jne   L
R:
   inchar ch1
   movzx cx, ch1
   cmp   cx, '.'
   je    ED
   cmp   cx, '0'
   jl    R                      2) jb - беззнаковые
   cmp   cx, '9'
   jg    R                      2) ja
   sub   cx,'0'
   test  cx,1
   jne   R
   cmp   ax,0                   3) ax НИЧЕМУ не равно !
   je    R2
   mul   cx                     3) ax НИЧЕМУ не равно !
   cwd                          4) Беззнаковые
                                5) И так после mul <dx,ax> !
   Jmp R
R2:
   movzx eax, ch1               6) НЕ понято условие "произведение"
                                   а НЕ СУММА
   Jmp R
ED:
   mov s,eax
   outword s
                                1) exit
end Start
```

ОЦЕНКА=1

4.

ОЦЕНКА=0

5.

ОЦЕНКА=0

107-№3. Бобровская А.С.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*G-№); G – Ваша группа
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для №**mod** 3=0; T=**dw** для №**mod** 3=1; T=**dd** для №**mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

5. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

107-№7. Карасев Г.И.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

2. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_3. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем P1, описанной в каком-то другом модуле. Если значение P1 при этом выйдет за допустимый диапазон, то обнулить его.

#1 T = dw

```
include console.inc
.data
   X dw ?
.code
Start:
    inint X
    xor   ecx, ecx
    movzx eax, X
    cmp   eax, 0
    jge   Skip
    neg   eax
Skip:
    mov   ebx, 10
Back:
    xor   edx, edx
    div   ebx
    test  edx,1
    jne   Skip_again; Нечётное
    inc   ecx
Skip_again:

    cmp   eax, 0
    jne   Back

    outword ecx
    exit
end Start
```

ОЦЕНКА=6

#2 n = 700

```
PMatr proc
       push ebp
       mov  ebp,esp; База
       push eax
       push ebx
       push ecx
       mov  ebx,[ebp+8]
       mov  ecx,[ebp+12]
       mov  eax, ecx
       add  eax, 1 ; n+1

   L:
       cmp  byte ptr [ebx], '0'
       jb   Net
       cmp  byte ptr [ebx], '9'
       ja   Net
       mov  byte ptr [ebx], 7;
```

№ = 7 в ASCII №7 – служебный символ, как его записать в матр? 1) Как 7

```
   Net:
       add  ebx,eax; следующий диагональный
       loop L

       pop ecx
       pop ebx
       pop eax
       pop ebp
       ret 8
PMatr endp

call _PMatr@0                                  2) Плохой вызов, где параметры ???
```

ОЦЕНКА=4

№3

```
.data
   extern P1:word
.code
   public Del_3
Del_3 proc
       push ebp
       mov  epb, esp
       push eax
       push ebx
       push edx

       mov  ax, P1
       sub  dx, dx
       mov  bx, 7
       imul bx
       jo   Poln
       jmp  End              1) НЕЛЬЗЯ
   Poln:
       mov  ax, 0
   End:
       mov  P1,ax

       pop edx
       pop ebx
       pop eax
       pop ebp
       ret
Del_3 endp
end
```

ОЦЕНКА=5

107-№8. Кожеуров П.С.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

3. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

4. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

5. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_5. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**dw**) с именем P2, описанной в каком-то другом модуле.

---

1)
1023(0)
511(1)
255(1)
127(1)
63(1)
31(1)
15(1)
7(1)
3(1)
1(1)
2046=0000 0111 1111 1110(2)
1111 1000 0000 0001
\+ 1
-2046=1111 1000 0000 0010(2) = F802(h) 1) НАОБОРОТ

ОЦЕНКА=3

---

2)

```
include console.inc
.code
Start:
  inint eax
  cmp   eax, 100
  jb    @F
  cmp   eax,    999
  ja    @F
  outstr  'ДА'
  jmp     con
@@:
  outstr   'НЕТ'
con:
  exit
  end     Start
```

ОЦЕНКА=6

3)

```
sum_mas proc
    push ebp
    mov  ebp, esp
    push eax
    push ebx
    push ecx
    push edx
    push esi
    mov  esi, [ebp+8]; ^A
    sub  ebx, ebx
    mov  ecx, 1
@@:
    mov     eax, [esi];
    mul     ecx

    add     ebx,  eax
    inc     ecx
    add     esi, 1
    cmp     cx, [ebp+12]
    ja      @B
    mov     [ebp+16], bx
    pop     esi
    pop     edx
    pop     ecx
    pop     ebx
    pop     eax
    pop     ebp
    ret 12
sum_mas endp

    push  offset Y
    movzx eax, N
    push  eax
    push  offset A
    call   _sum_mas@0
```

1) Это (A[1],A[2],A[3],A[4]) ???
2) НЕ понято условие
"в формате *байта*"

ОЦЕНКА=0

4)

```
Sum_1 macro X:req
      local B,L
      B=0
if type X NE 1 OR type X NE 2 OR type X NE 4
      B=1
endif
for reg, <al,ah,bl,bh,cl,ch,dl,dh,ax,bx,cx,dx,\
          eax,ebx,ecx,edx,ebp,esp,esi,edi>
  ifidni   <reg>,<X>
      B=1
      exit
  endif
endm
if B EQ 1
     .err   <bad input>
      exitm
endif
if type X EQ 4
      mov   edx, X
else
```

1) ВСЕГДА **true** !

2) **exitm** !!!

```
      movzx edx, X
endif
      sub   ax, ax
      mov   ecx, 32
L:
      shl   edx, 1
      adc   ah, 0; ah:=Число 1
      loop  L;
;      mov   al, 32
      sub   al, ah                          2) ПРОСТО sub al,32 ???
;      B = type X
;      B = 4 - B
;      B = B * 8
      sub   al, B                           3) ПРОСТО sub al,8*(type X)
      Endm
```

ОЦЕНКА=4

---

5)

```
.code
    extern P2: dword            1) НЕТ, P2=word !!
    public Del_5
.code
Del_5 proc
    push ebp
    mov  ebp, esp
    push eax
    push ecx
    sub  eax, eax
    mov  ax, P2
    mov  cx, 8
    div  cx                     2) ПЛОХОЕ ДЕЛЕНИЕ dx=???
    sub  ah, ah                 3) НЕТ, ответ в ax
    mov  P2, ax
    pop  ecx
    pop  eax
    pop  ebp
    ret
Del_5 endp
    end
```

ОЦЕНКА=1

107-№10. Козуб Д.В.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

```
001      ВВЦ    100    200    000    read(x)
2        МОД    000    100    019    <0> :=x[1] mod 2
3        ВЧЦ    000    000    017    <0> - 0                          1) НЕ НАДО
4        УСЛ    005    007    007    нечет → дальше проверка
5        ВЫЦ    018    001    000    вывод 1                          2) goto 015
6        БЕЗ    000    016    000    стоп
7        ВЧЦ    000    100    299    x[1] - x[200]
8        УСЛ    009    015    015    не равны → вывод 1
9        ВЧЦ    020    020    018    dec(n)
010      УСЛ    011    013    013    конец?
1        ВЫЦ    017    001    000    вывод 0
2        БЕЗ    000    016    000    стоп
3        СЛЦ    007    007    021    inc(i), inc(j)
4        БЕЗ    000    007    000    цикл
5        ВЫЦ    018    001    000    вывод 1
6        СТОП   000    000    000
7        00     000    000    000    конст 0
8        00     000    000    001    конст 1
9        00     000    000    002    конст 2
020      00     000    000    100    n = 100
1        00     000    000    511    дельта
```

ОЦЕНКА=6

---

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1(2) количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

```
include console.inc
.const
message1 db 'Да'
message2 db 'Нет'
.data
X dw ?
.code
start:
   inint X
   mov   ax, X
   cmp   ax, 10
   jb    No
   cmp   ax, 99
   ja    No
   outstr offset message1
   jmp finish
No:
   outstr offset message2
finish:
   exit
   end start
```

ОЦЕНКА=6

---

3. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

```
Sum_1 macro X:Req
   local K
ifb <X>                                 1) НЕ БЫВАЕТ !!!
   echo No argument
   .err
   exitm
endif
   K = type X
if K NE 1 or K NE 2 or K NE 4           2) ВСЕГДА true !
   echo wrong argument
   .err
   exitm
endif
for reg, <al, ah, bl, bh, cl, ch, dl, dh, ax, bx, \
          cx, dx, dx, si, di, bp, sp, eax, ebx, ecx, edx, esi, edi, ebp, esp>
  ifidni <reg>, <X>
   echo wrong argument
   .err
   exitm
  endif
endm
if K EQ 1
   push ecx
   mov  cl, X
   sub  al, al
@@:                                     3) Плохо в М/определении, понять или спросить
   shl  cl, 1
   adc  al, 0
   test cl, cl
   jnz  @B
   neg  al
   add  al, 8
   pop  ecx
endif
if K EQ 2
   push ecx
   mov  cx, X
   sub  ax, ax
@@:
   shl  cx, 1
   adc  ax, 0
   test cx, cx
   jnz  @B
   neg  ax
   add  ax, 16
   pop  ecx
endif
if K EQ 4
   push ecx
   mov  ecx, X
   sub  eax, eax
@@:
   shl  ecx, 1
   adc  eax, 0
   test ecx, ecx
   jnz  @B
   neg  eax
   add  eax, 32
   pop  ecx
                                        else "Плохой тип X" !!!
endif

   endm
```

ОЦЕНКА=3

4. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем `Del_5`. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**`dw`**) с именем `P2`, описанной в каком-то другом модуле.

```
include console.inc
   extrn P2: word
                              1) public Del_5 !!!
Del_5 proc
   push eax
   push ebx
   push edx

   mov ax, P2
   cwd                        2) НЕТ, беззнаковое
   mov bx, 10
   div bx
   mov P2, ax

   pop edx
   pop ebx
   pop eax
Del_5 endp
   end
```

ОЦЕНКА=2

107-№12. Москалев А.Р.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

107-№13. Мурин Е.А.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*107-13); G – Ваша группа
0FBDFh                                  1) Наоборот 0DFFBh
```

ОЦЕНКА=3

---

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

```
    include console.inc
.data
      X dw ?
.code
start:
      xor  ecx, ecx
      ININT X
      cmp  X, 0
      je endofprog
      mov bx, 10
circle :
      mov  ax, X
      idiv bx                       1) Плохое деление dx=?
      test bx, 1                    2) НЕТ, цифра в dx !!!
      jnz  next
      inc  ecx
next :
      cbw                           3) УЧИТЬ ДЕЛЕНИЕ !!!
      cmp ax, 0
      jne circle
      OUTWORD ecx
endofprog :
      cmp ecx, 0
      je zero
      jmp skip
zero :
      OUTWORD 1
skip :
      exit
      end start
```

ОЦЕНКА=0

---

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных 8, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

```
      include console.inc
.data
      t db 0
.code
start:
      xor ebx, ebx
      xor ecx, ecx
circle :
      INCHAR bh
      cmp bh, '.'
      je  endofprog
      cmp bh, '*'
      jne next
```

```
      mov t, 1
next :
      cmp t, 1
      jne circle
      xor eax, eax
      cmp bh, '0'          1) Зачем? Просто cmp bh, '8'
      jb  circle
      cmp bh, '9'
      ja  circle
      sub bh, '0'
      movzx eax, bh
read :                     2) Зациклились, нет проверки на точку!
      xor ebx, ebx
      INCHAR bh
      cmp bh, '0'
      jb check
      cmp bh, '9'
      ja check
      sub bh, '0'
      mov edx, 10
      mul edx ;нас не интересует что будет на edx
      add eax, ebx
      jmp read             3) Нет проверки на точку, когда одни цифры!
check :
      OUTWORD eax          4) Этого НЕТ в условии !!!
      test eax, 111b
      jnz skip
      inc ecx
skip :
      cmp bh, '.'
      jne circle
endofprog :
      OUTWORD ecx
      exit
      end start
```

ОЦЕНКА=2

---

4. Написать макроопределение с заголовком

`First_1` **`macro`** `X:Req`

Оно должно для своего параметра `X`, который может быть только форматов `m8`, `m16` или `m32`, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении `X`, например, для `X` **`db`** `10101010b` необходимо получить `X` **`db`** `11110000b`. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

```
   .686
.model flat
.data
.code
First_1 macro X:req
if tipe X EQ 0             1) type
     .err                  2) exitm - понять
endif
if tipe X EQ 1             3) Не понято условие форматов m8, m16 или m32
     mov ah, X
circle :
     cmp ah, 0
     je  endc
     mov bh, ah
     and bh, 1
     cmp bh, 0
```

```
      je  circle
      shl dh, 1
      add dh, 1
      jmp circle
endc :
      cmp dh, 0
      je endX
      test dh, 10000000b
endX :
      mov X, dh
endif
      endm
start:
exit
end start
```

ОЦЕНКА=0

---

5. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем `Del_3`. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**`dw`**) с именем `P1`, описанной в каком-то другом модуле. Если значение `Perem3` при этом выйдет за допустимый диапазон, то обнулить его.

```
    .686
.model flat
.data                          1) Не понято условие описанной в каком-то другом модуле.
.code
      public Del_3
Del_3 proc
      push ebp
      mov  ebp, esp
      push eax
      push ebx
      push edx
      mov  ebx, [ebp + 8]      2) Не понято условие без параметров
      mov ax, [ebx]
      mov dx, 13
      imul dx
      cmp dx, 0
      je endofproc
      mov ax, 0
endofproc :
      mov [ebx], ax
      pop edx
      pop ebx
      pop eax
      pop ebp
      ret 4
Del_3 endp
end
```

ОЦЕНКА=0

107-№14. Назаренко Г.С.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".
2. Написать макроопределение с заголовком

Sum_1 **macro** X:Req

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.
3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_5. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**dw**) с именем P2, описанной в каком-то другом модуле.

---

1) Задание

```
include console.inc
.data
x dd ?
.code
start:
        xor   edi,edi
        mov   ebx,10
        inint x
L1:
        cmp   x,0
        jz    Outgo
        mov   eax,x
        xor   edx,edx
        div   ebx
        mov   dword ptr x,eax          1) Не надо dword ptr
        inc   edi
        jmp   L1
Outgo:
        ConsoleMode                    2) Зачем ???
        cmp   edi,3
        jne   Nope
        outstr 'ДА'
        jmp   Outik
Nope:
        outstr 'НЕТ'
Outik:
        ConsoleMode                    1) Зачем ???
        exit
        end start
```

ОЦЕНКА=6

==========

2) Задание

```
include console.inc
Sum_1 macro x:req
        local Cycle,K,Next
        K=0
for i,<al,ah,bl,bh,cl,ch,dl,dh \
```

```
      ax,bx,cx,dx,si,di,bp,sp \
      eax,ebx,ecx,edx,esi,edi,ebp,esp>
  ifidni <i>,<x>
            K=1
            exitm
  endif
endm
if type x EQ 4 and K EQ 0
            mov ecx,32
            mov ebx,x
elseif type x EQ 2 and K EQ 0
            mov ecx,16
            movzx ebx,x
elseif type x EQ 1 and K EQ 0
            mov ecx,8
            movzx ebx,x
else
            echo 'Bad input x'
            .err
            exitm
endif
            mov al,0
Cycle:
            shr ebx,1
            jc Next
            add al,1
Next:
            Loop Cycle
            endm
.data
.code
start:

       exit
end start
```

ОЦЕНКА=6

==========

3) Задание

```
.686
.model flat
            extrn p2:word
.code
            Public Del_5
Del_5 proc
            push  eax
            push  ebx
            mov   bh,14
            mov   ax,p2
            div   bh                    1) Надо ДЛИННОЕ деление, понять или спросить
            movzx ax,al
            mov   p2,ax
            pop   ebx
            pop   eax
```

```
        ret 0
Del_5 endp
        end
```

ОЦЕНКА=3

==========

107-№16. Петров П.В.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

5. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

№1

001:ВВЦ 100 200 000
002:МОД 100 ??? 000
003:УСЛ ??? 004 000
004:ПЕР 102 000 013
005:ВЧЦ 000 102 100
006:ПБ 000 011 000
007:
008:

ОЦЕНКА=0

№2

1111 1000 0000 0010 1) Перевёрнутое !

ОЦЕНКА=3

№3

```
include console.inc
.data
   ch dw ?
   da db "ДА"
   net db "НЕТ"
   N  db ?
   TT db ?
.code
N  equ 16                  1) НЕЛЬЗЯ, имя уже ОПИСАНО
TT equ 2                   1) НЕЛЬЗЯ, имя уже ОПИСАНО
   inint ch
   mov   eax, ch           2) НЕЛЬЗЯ dd<>dw
   xor   ecx, ecx
   mov   ecx, N            3) НЕЛЬЗЯ dd<>db
RPT:
   shl  eax, 1
   cmp  cf, 1              4) НЕТ ТАКОГО !!!
   je   NEXT
   loop RPT                5) Не понято условие "в десятичной записи"
NEXT:    cmp ecx, TT
```

```
   jne NET
   outstr da
   jmp FIN
NET:    outstr net
FIN:     exit
   end
```

ОЦЕНКА=0

№4 НЕТ

ОЦЕНКА=0

№5

Ассесблер вызовет так

```
   push offset Y
   push N
   push offset A
   call SUM
```

```
SUM proc
    push ebp
    mov ebp, esp
    push ebx
    push esi
    push edi
                                                0)    push eax !!
; dword[ebp + 8] - адрес массива A
; dword[ebp + 12] - длина массива N
; dword[ebp + 16] - адрес переменной Y
    mov ebx, dword[ebp + 12]; N                 1) dword ptr и НЕ НАДО
    mov esi, dword[ebp + 8]; ^A                 1) dword ptr и НЕ НАДО
    dec esi
    xor edi, edi
loops:
    cmp ebx, 0
    je  end                                     2) Учить цикл Loop
    movsx eax, byte[esi + ebx]                  3) ЭТО сразу 4 элемента !!!
    imul  eax, ebx
    add   edi, eax
    dec   ebx
    jmp   loops
end:                                            4) НЕЛЬЗЯ, служебное слово
    mov dword[ebp + 16], edi                    5) ЭТО АДРЕС(Y):=edi,
                                                   А надо Y:=edi
    pop edi
    pop esi
    pop ebx
    mov esp, ebp
    pop ebp
    xor eax, eax                                6) Это НЕ функция !!!
    ret 12
SUM endp
```

ОЦЕНКА=0

107-№17. Семенов П.А.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*G-№); G – Ваша группа
```

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

4. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

1. НЕТ => ОЦЕНКА=0

2.

```
include console.inc
.data
   X dd ?
.code
start:
   inint X
   mov  eax, X
   xor  ecx, ecx
program_loop:
   mov  ebx, 10                          1) Зачем это в ЦИКЛЕ ?
   xor  edx, edx                         2) НЕТ, знаковое !!!
   idiv ebx;  div->eax, mod->edx
   cmp  edx, 2                           3) Просто test edx,1
   je   if_even
   cmp  edx, 4                           4) А если edx=-4 ???
   je   if_even
   cmp  edx, 6
   je   if_even
   cmp  edx, 8
   je   if_even
   cmp  edx, 0
   je   if_even
   jmp  continue_loop

if_even:
   add  ecx, 1
   jmp  continue_loop                    2) ?????

continue_loop:
   cmp  eax, 0
   jne  program_loop
   outword ecx
exit
end start
```

ОЦЕНКА=2

3.

```
include console.inc
.code
start:
   xor ecx, ecx
start_loop:
   inchar bl
   cmp bl, '*'
   je  if_zvezda
   cmp bl, '.'
   jne start_loop
   outword ecx
   exit
if_zvezda:
   inchar bl
   cmp bl, '5'; if 0 nothing changes
   je  if_five
   cmp bl, '.'
   jne if_zvezda
   outword ecx
   exit
if_five:
   add ecx, 5
   jmp if_zvezda

end start
```

ОЦЕНКА=6

4.

```
proc F
   push ebp
   mov  ebp, esp
   push eax
   mov  eax, [ebp + 8]; var MAS
   push ecx
   mov  ecx, [ebp + 12]; N
   push ebx
   xor  ebx, ebx
start_loop:
   mov [eax], 17          1) mov byte ptr [eax], 17
                          2) Не понято условие "символы цифр"
   add eax, ecx
   add eax, 1
   add ebx, 1
   cmp ebx, ecx
   jne start_loop

   pop ebx
   pop ecx
   pop eax
   pop ebp
endp F

   push N
   push offset X; X = MAS
   call F
```

ОЦЕНКА=1

---

1. –(1070-17)=-1053

1053 = 1024 + 16 + 8 + 4 + 1
1053 = 0100 0001 1101
-1053 = 1..1 1011 1110 0011 1) Наоборот, перевёрнутое !

ОЦЕНКА=3

107-№20. Сычев П.С.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

№2

Nomer 20 => dd => Nomer mod 3 + 1= (3)

```
include console.inc
.data
  x dd ?
.code
start:
      inint x
      mov eax, x
      mov ebx, 10
      sub ecx, ecx
L:
      sub edx, edx
      div ebx
      inc ecx
O:
      cmp eax, 0
      jnz L
      cmp ecx, 3
      je M
      outstr 'НЕТ'
      jmp konec
M:
      outstr 'ДА'
konec:
      exit
      end start
```

ОЦЕНКА=6

№3

20 mod 6 + 2=4

```
include console.inc
.data
  S dq 0
.code
start:
      inchar bl
      cmp bl, '&'
      jne start           1)ЗАЦИКЛИЛИСЬ, нет проверки точки
M:
      inchar bl
      cmp bl, '0'
      jb  L
      cmp bl, '9'
      ja  L
      mov bh, bl
      cmp bh, '4'
      je  N
      cmp bh, '8'
      je  N
      cmp bh, '0'
      je  N
      jmp L

N:
      sub bh, '0'
      movzx eax, bh
      add dword ptr S, eax
      adc dword ptr S+4,0
L:
      cmp bl, '&'               2) НЕ понято условие
                                   " вводит текст до точки "
      jnz M
      outword S

          exit
end start
```

ОЦЕНКА=0

№4

```
include console.inc
.data
N equ 10
   A db 1,2,3,4,5,6,7,8,9,0
   Y dw ?
.code
Sum proc
   push ebp
   mov  ebp, esp
   push ecx
   push ebx
   push eax
   push edi
   push edx
   mov  edi, 0; N
   mov  ebx, [ebp+8]; ^A
   mov  cx, 0

L: cmp  edi, [ebp+16]                  1) Учить цикл Loop
   jz   E
   movzx ax, byte ptr [ebx][edi]       2) ax:=(2 элемента сразу)
   inc   edi
   mul   di                            3) Не понято условие
```

```
                                        1*A[1]+2*A[2]+...
   add  cx, ax
   jmp  L
E:
   mov ebx, [ebp+12]
   mov [ebx], cx
   pop edx
   pop edi
   pop eax
   pop ebx
   pop ecx
   pop ebp
   ret 12
Sum endp
Start:

   mov eax, N;
   push eax
   push offset Y
   push offset A
   call Sum
   outint Y
   exit
end Start
```

ОЦЕНКА=2

№5

```
Sum_1 macro X:req
      local K, L, N
      K=0
for reg,<eax,ebx,ecx,edx,esi,edi,ebp,esp,
         ax,bx,cx,dx,si,di,bp,sp,
         ah,al,bh,bl,ch,cl,dh,dl>
  ifidni <reg>,<X>
       K=1
       exitm
  endif
endm
if K EQ 1 and type X EQ 1
       mov ebx, X              1) НЕЛЬЗЯ ebx=dd <> X=db
       mov ecx, 8
       mov al, 0
elseif K EQ 1 and type X EQ 2
        mov ebx, X             2) НЕЛЬЗЯ ebx=dd <> X=dw
        mov ecx, 16
        mov al, 0
elseif K EQ 1 and type X EQ 4
        mov ebx, X
        mov ecx, 32
        mov al, 0
else
        .err <WRONG ARGUMENT IN MACROS Sum_1>

                                3) exitm !!!
endif
  endif                         4) НЕТ if !!!
L:
        shl ebx                 5) ,1 !!!
        jnz N                   6) ВСЕГДА на N !!!!
                                7) Надо проверять CF !
N:
```

```
        inc al
        dec ecx
        jnz L
        endm
```

ОЦЕНКА=1

№1

1	ввц	100	200	000	
2	мод	000	100	012	
3	пр	000	007	000	
4	слц				1) ???
5					
6					
7	слц	002	002	013	
8	слц				
9					
10	выц	000	002	000	
11	стоп	000	000	000	
12	00	000	000	002	
13	00	000	000	001	

ОЦЕНКА=0

107-№21. Трофимов А.Д.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*G{107}-№{21}); G – Ваша группа
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных №{21} **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

5. Пусть на Паскале дано описание типа массива:

```
const n=100*№{21}; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№{21}). Привести пример вызова этой процедуры.

Задача 5.

Вызов:

```
      Mov ebx, 2100
      Push ebx              1) Push 2100
      Push offset Mas
      Call Modif
```

Процедура:

```
Modif proc
      Push ebp
      Mov  ebp, esp
      Push ebx
      Push ecx
      Push eax
      Mov  ecx, [ebp+12]
      Mov  ebx, [ebp+8]
      Mov  eax, 2101              1) Откуда процедура ЗНАЕТ число 2101 ?
      Dec  ecx
L:    mov  dh, byte ptr [ebx]     2) Не надо byte ptr
      Cmp  dh, '0'
      Jb   L1
      Cmp  dh, '9'
      Ja   L1
      Mov  bytr ptr [ebx], 21

L1:   add ebx, eax
      Loop L

      Pop eax
      Pop ecx
      Pop ebx
      Pop ebp
      Ret 8
Modif endp
```

ОЦЕНКА=3

Задача 3.

```
   Include console.inc
   .data?
     X db ?
   .code
```

```
Start:
   Xor edx, edx
   Mov dl, 10
   Inint X
   Xor eax, eax
   Mov al, X
   Cmp al, 0
   Jge L
   Neg al
   Cmp al, 0
   Jnz S
   Inc dh; считаем, что для нуля – ответ – одна цифра
   Jmp K
S: Div dl
   Cmp ah, 0              1) Просто test ah,1
   Jz  L
   Cmp ah, 2
   Jz  L
   Cmp ah, 4
   Jz  L
   Cmp ah, 6
   Jz L
   Cmp ah, 8
   Jz  L
   Jmp S1                 2) НЕТ S1 !!!
L: inc dh
   Cmp al, 0
   Jz K
   Xor ah, ah
   Jmp S

K: outword dh
                          3) exit
   End Start
```

ОЦЕНКА=4

Задача 2.

-1049 = 1111 1011 1110 0111 – двоичный вид
(если тут это имеется в виду, то в памяти оно хранится в обратном порядке)
ОЦЕНКА=6

Задача 4.

```
Incude console.inc
.code
Start:
   Xor dh, dh
   Xor ebx, ebx
L: Xor eax, eax
   inchar al
   Cmp al, '.'
   Jz  K
   Cmp dh, 0                  1) dh ВСЕГДА=0 !!!
   Jz  L                      2) Нет анализа '*' !!!
   Cmp al, '*'
   Mov dh, 1                  3) Mov dh, 1 НЕЗАВМСИМО от '*' !!!
   Jz  L
   Cmp al, '0'
   Jnz L
   Sub al, '0'
```

```
   Add ebx, eax
   Cmp al, '2'
   Jnz L
   Sub al, '0'
   Add ebx, eax
   Cmp al, '4'
   Jnz L
   Sub al, '0'
   Add ebx, eax
   Cmp al, '6'
   Jnz L
   Sub al, '0'
   Add ebx, eax
   Cmp al, '8'
   Jnz L
   Sub al, '0'
   Add ebx, eax
   Jmp L
K: outword ebx
                                        4) exit
   End Start
```

ОЦЕНКА=1

107-№22. Утехина М.П.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

2. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

3. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

4. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_5. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**dw**) с именем P2, описанной в каком-то другом модуле.

Задание 1.
0000 0111 1111 1110
1111 1000 0000 0001
Ответ : 1111 1000 0000 0010 1) Наоборот

ОЦЕНКА=3

Задание 2

```
include console.inc
.code
Start:
      sub ecx, ecx; символ
      sub eax, eax; сумма
      sub ebx, ebx
Wh:
      inchar ch
      cmp ch, '.'
      je  Exi
      cmp ebx, 1
      je  L
      cmp ch, '&'
      jne Wh
      mov ebx, 1
L:
      cmp ch, '0'
      jb  Wh
      cmp ch, '9'
      ja  Wh
;если это цифра
;     mov dh, '0'
;     sub ch, dh
;     cmp ch, 6              1) ПРОСТО cmp ch, '6'  !!!
      je  Plus
      jmp Wh
Plus:
      add al, ch             2) Плохая сумма, надо Longword !
      jmp Wh
Exi:
      outword ax             3) НЕТ, eax
        exit
        end Start
```

ОЦЕНКА=3

Задание 3

```
Sum_1 macro X:Req
      local K, L
      K=0
for reg,<al,ah,bl,bh,cl,ch,dl,dh \
            ax,bx,cx,dx,si,di,bp,sp \
            eax,ebx,ecx,edx,esi,edi,ebp,esp>
  ifidni <reg>,<X>
      K=1
      exitm
  endif
endm
      push ecx
if type X EQ 0 or K EQ 1
      .err <Bad arg in macros>
      exitm
elseif type X EQ 1
      movzx ecx, X
      mov ah, 8
elseif type X EQ 2
      movzx ecx, X
      mov ah, 16
elseif type X EQ 4
      mov ecx, X
      mov ah, 32
endif
      sub al, al
L:
      sub ah, 1
      shr ecx, 1
      jc  L; если бит 1 , то прыг
      add al, 1
      cmp ah, 0
      jne L
      pop ecx
endm
```

ОЦЕНКА=6

Задание 4

```
.data
      extrn P2: dword; внешняя переменная      1) word ПО УСЛОВИЮ
.code
      public Del_5
Del_5 proc
      push eax
      sub  eax,eax
      mov  ax, P2
      div  22                                  2) УЧИТЬ ДЕЛЕНИЕ !
      sub  ah,ah                               3) Надо ДЛИННОЕ деление
      mov  P2, ax
      pop eax
      ret
Del_5 endp
end
```

ОЦЕНКА=2

107-№6. Зиннуров А.Р.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

2. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_5. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**dw**) с именем P2, описанной в каком-то другом модуле.

№1
|-2046| = 2046 (в 10 с/сч) = 0000011111111110(в 2 с/сч)
0000011111111110 – прямой код
1111100000000001 – обратный код
+ 1
1111100000000010 – дополнительный код
В машинном представлении обратный порядок
Ответ: 0000 0010 1111 1000

ОЦЕНКА=6

№2

```
Sum_1  macro X:req
       Local K, T, Chikl, Next
       K = type X
       T = 0
if K EQ 1 or K EQ 2 or K EQ 4
  for reg, <al,ah,bl,bh,cl,ch,dl,dh \
            ax,bx,cx,dx,si,di,bp,sp \
            eax,ebx,ecx,edx,esi,edi,ebp,esp>
    ifidni <reg>, <X>
       T = 1 ;; X - регистр
       exitm
    endif
  endm
else
       .err <Bad type argument>
       exitm
endif
if T EQ 0 ;; X - не регистр
  if K EQ 1 or K EQ 2
       movzx eax, X ;; eax := Longword(X) (беззнаково)
  else
       mov eax, X ;; eax := X
  endif
       sub ecx, ecx ;; ecx := 0 - сумма '1'
Chikl:
       mov ebx, eax
       and ebx, 1
       cmp ebx, 1
       jne Next
       add ecx, 1
Next:
       shr eax                                          1) ???????
       cmp eax, 0
       jne Chikl
;      mov eax, type X ;; eax := type X (= 1, 2, 4)
;      mov edx, 8 ;; edx := кол-во битов в байте
;      mul edx ;; eax := type X * 8 (= 8, 16, 32)       2) mov eax,8*(type X)

       sub eax, ecx ;; eax := type X * 8 - кол-во '1' = кол-ву '0'
else
       .err <Argument - registr> ;;
```

```
; подразумевается, что сразу выйдет за endm, поэтому нет exitm
endif
       endm
```

ОЦЕНКА=6-

№3

```
include console.inc
.code
       extern P2:word
       public Del_5
.code
Del_5 proc
; Пролог
       push ebp
       mov  ebp, esp
       push eax
       push ebx
       push edx
; Тело процедуры
       movzx eax, P2 ; eax := Longword(P2)
       sub edx, edx ; edx := 0
       mov ebx, 6 ; ebx := 6 (мой номер)
       div ebx
       mov P2, ax ; P2 := ax
; Эпилог
       pop edx
       pop ebx
       pop eax
       pop ebp
       ret
Del_5 endp
       end
```

ОЦЕНКА=6

108-№2. Батушин А.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

5. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

Задача 3:

```
include console.inc ; № mod 3 + 1 = 2 + 1 = 3
.data
  X dd ?
.code
Start:
  ClrScr
  inint X
  mov eax, 100
  mov ebx, 999
  cmp eax, X
  ja  E
  cmp ebx, X
  jb  E           1) jbe
  outstrln "ДА"
  jmp R
E:
  outstrln "НЕТ"
R:
  exit
  end Start
```

ОЦЕНКА=5

Задача 5:

```
include console.inc
.data
N equ 200
A db N dup (?)
Y dd ?
.code
SUM proc
  push ebp
  mov  ebp, esp
  push eax
```

```
  push ebx
  push ecx
  push edx
  mov  eax, [ebp + 16] ; offset A (db)  1) Не тот порядок
  mov ebx, [ebp + 12] ; N (4 bytes)
  mov edx, [ebp + 8] ; offset Y (dd)
ADDING:
  mov ecx, ebx; N
SUMMA:
  add [edx],[eax][ebx]                  2) НЕЛЬЗЯ Память-память
                                        3) Где 1*A[1]+2*A[2]+...
  loop SUMMA                            4) Не понято условие задачи
  sub ebx, 1
  cmp ebx, 0
  jne ADDING
  pop edx
  pop ecx
  pop ebx
  pop eax
  pop ebp
  ret 12
SUM endp

  push offset A         1) Не тот порядок
  push N
  push offset Y
  call SUM
```

ОЦЕНКА=0

Задача 4:

```
include console.inc
; Поскольку № mod 6 + 2 = 2 + 2 = 4, то ищем кратные 4 цифры - 4 и
8
.data
  s dd 0
.code
Start:
  ClrScr
VVOD:
  inchar dl
  cmp dl, '&'
  je  PRAV_VVOD
  cmp dl,'.'
  je OUTPUT
  jmp VVOD
PRAV_VVOD:
  inchar dl
  cmp dl, '4'
  je  SUM_4
  cmp dl, '8'
  je SUM_8
  cmp dl, '.'
  je OUTPUT
  jmp PRAV_VVOD
SUM_4:
  add s, 4
```

```
  jmp PRAV_VVOD
SUM_8:
  add s, 8
  jmp PRAV_VVOD
OUTPUT:
  outword s
                                1) exit
  end Start
```

Задача 2:
Число: X dw -2046 (2 байта) Двоичная
Прямой код: 10000111 11111110
Обратный код: 11111000 00000001
Дополнительный код: 11111000 00000010
Тогда X представляется в двоичнов виде: (обратный порядок)
00000010
11111000

ОЦЕНКА=6

Задача 1:

ОЦЕНКА=0

108-№4. Денисов В.М.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046


прямой код : 0000 0111 1111 1110
обратный   : 1111 1000 0000 0001
доп        : 1111 1000 0000 0010
              F    8    0    2
Ответ: 02F8h
```

ОЦЕНКА=6

---

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

```
;Number = 4    => 2 знач. цифры
include console.inc
.data
   X dw ?
.code
Start:
   inint X
   mov ax,X
   mov bl,10
   div bl                  1) Надо ДЛИННОЕ деление
   cmp al,9
   jg  L                   2) ja - беззнаковые
   cmp al,1
   jl  L                   2) jb - беззнаковые
   outstr “Да”
   exit
L:
   outstr “Нет”
   exit
   end Start
```

ОЦЕНКА=3

---

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

```
   ;number =  4
include console.inc
.code
Start:
      mov ebx,0            1) sub ebx,ebx
L:
      inchar al
      cmp al, ‘.’
      je F
      cmp al,’&’
      jne L
M:
      inchar al
      cmp al,’.’
      je F
      cmp al,’6’
```

```
      jne M
      add ebx,6
                                  2) jmp !!!
F :   outword ebx
                                  3) exit
end Start
```

ОЦЕНКА=4

---

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива `A` целых беззнаковых чисел в формате *байта*, длину этого массива `N≥300` (массив считается пронумерованным от `1` до `N`) и адрес целочисленной переменной `Y` в формате слова. Процедура возвращает в `Y` сумму `1*A[1]+2*A[2]+... N*A[N]`. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

```
    Include console.inc
    .data
    A db 300 dup (?)
    Y dw ?
    .code
    Summa proc
       push ebp
       mov  ebp, esp
       push edx
       push ecx
       push ebx
       push eax
       push esi
       push edi
       mov  ecx, [ebp + 12]; N
       mov  esi, [ebp+16];   ^Y            1) НЕ ТОТ порядок !!!
       mov  edi, [ebp+8];    ^A
       mov  bx,0
       mov  dx,1
L:
       mov  al,[esi]
       cdw                              2) НЕТ, беззнаковые
       mul  dx
       add  dx,1
       add  esi,4                       3) Не понято условие " в формате байта "
       loop L
       mov word ptr [edi],bx            1) НЕ ТОТ порядок !!!
                                        4) Не надо word ptr
                                        5) В bx всегда 0
       pop  edi
       pop  esi
       pop  eax
       pop  ebx
       pop  ecx
       pop  edx
       pop  ebp
       ret 12
Summa enp

Start:
       push offset A                    6) НЕ ТОТ порядок !!!
       push 100                         6) По условию N≥300
       push offset Y
       call Summa
       outintln Y
       exit
end Start
```

ОЦЕНКА=2

---

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для `X db 10101010b` должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

```
Sum_1 macro X:req
      local L1, En
      push ebx
      push edx
for reg,<al,ah,bl,bh,cl,ch,dl,dh \
         ax,bx,cx,dx,si,di,bp,sp          \
         eax,ebx,ecx,edx,esi,edi,ebp,esp>
  ifidni <reg>,<X>
      .err <Bad arguments: registers>
      exitm
  endif
endm
if type X EQ 1 or type X EQ 2
      movsx ebx, X
else if type X EQ 4
      mov  ebx, X
else
      .err <Bad arguments: constants>
      exitm
endif
      xor  al, al
      mov  edx, 1
L1:
      cmp  edx, 0
      je   En                        1) Цикл ВСЕГДА на 32
                                        Но НУЛЕЙ в байте МЕНЬШЕ, чем в слове !
      shl  edx, 1
      test ebx, edx
      jnz L1
      inc al
      jmp L1
En:
      pop edx
      pop ebx
      endm
```

ОЦЕНКА=2

108-№22. Элишаков Д.Г.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_5. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**dw**) с именем P2, описанной в каком-то другом модуле.

Задача 1

```
1  ВВЦ 100 200 000 ;ввод массива
2  МОД 100 100 015 ; X[1] чёт/нечёт
3  ПР  000 010 000
4  ВЫЦ 000 100 299
5  ПНР 000 010 000
6  СЛЦ 004 004 013
7  ВЧЦ 000 004 014 ;проверка середины
8  ПМ  000 004 000
9  ВЧЦ 012 012 012
10 ВЫЦ 012 001 000
11 СТОП 000 000 000
12 00  000 000 001 ;const 1
13 00  000 000 511
14 ВЫЦ 000 200 199 ;проверка середины
15 00 000 000 002 ;const 2
```

ОЦЕНКА=6

Задача 2

```
include console.inc

.data
   T dw ?
.code
Start:
   inint T
   movzx eax, T
   mov   ebx, 10
   sub   ecx, ecx
Cycle:
   xor   edx, edx
   div   ebx
   inc   ecx
   cmp   eax, 0
   jne   Cycle
   cmp   ecx, 2
   jne   No
   outstr 'YES'
   jmp   Fin
No:
   outstr 'NO'
Fin:
   exit
   end Start
```

ОЦЕНКА=6

Задача 3

```
include settings.inc
include io2020.inc

   extern P2: word;

.code

Del_5 proc
    push eax
    push ebx
    push edx
    mov  dx, 0
    mov  ax, P2
    mov  bx, 22
    div  bx
    mov  P2, ax
    pop edx
    pop ebx
    pop eax
    ret
Del_5 endp
   end
```

```
1) include console.inc
2) НЕ надо !
   Как со сдачей программ !?


3) public Del_5   !!!
```

ОЦЕНКА=2

108-№3. Гилязова А.Ф.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

```
include console.inc
.data
   X db ?
.code
start:
   inint X
   xor ecx, ecx
   cmp X, 0                        1) А если не существует ???
   jge Next
   neg X ;для отрицательных
Next:
   mov bl, 10
   mov al, X
L:
   cmp al, 0
   je  fin
   test al, 1
   jnz Skip ; если не нечетное уходим
   inc ecx
Skip:
   mov ah, 0 ;для деления
   div bl
   jmp L
fin:
   outword ecx                     2) Для X=0 надо ответ 1 !!!
   exit
   end start
```

ОЦЕНКА=5

---

2. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

```
First_1 macro X:Req
   local L1, L2, L3,K1, K2, K3, fin
if type X ne 1 or type X ne 2 or type X ne 4
   echo Wrong type_ of X
   .err <wrong type_ of X>
   exitm
                                        1) endif
if type X EQ 1
         mov bl, X
         xor eax, eax
         mov ecx, 8;
L1:
         shr bl, 1
         adc al, 0; посчитали количество «1»
         Loop L1
         mov X, 0
         xor ecx, ecx
         mov cl, al
         cmp cl, 0
         je fin ;если нет единиц то число 0
```

```
K1:
        shr X, 1   ;сдвигаем и пишем слева единицы
        or X, 80h ;10000000
        loop K1

elseif type X EQ 2
        mov bx, X
        xor eax, eax
        mov ecx, 16
L2:                                     2) L2=L1=L3 Понять или спросить
        shr bx, 1
        adc al, 0
        Loop L2

        mov X, 0
        xor ecx, ecx
        mov cl, al
        cmp cl, 0
        je fin
  K2:                                   2) K2=K1=K3 Понять или спросить
        shr X, 1
        or X, 8000h
        loop K2
elseif type X EQ 4
        mov ebx, X
        xor eax, eax
        mov ecx, 32
  L3:
        shr ebx, 1
        adc al, 0
        Loop L3

        mov X, 0
        xor ecx, ecx
        mov cl, al
        cmp cl, 0
        je fin
  K3:
        shr X, 1
        or X, 80000000h
        loop K3
endif
fin:
        endm
```

ОЦЕНКА=5

---

3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем `Del_3`. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем `P1`, описанной в каком-то другом модуле. Если значение `Perem3` при этом выйдет за допустимый диапазон, то обнулить его.

(насколько я поняла P1 и Perem3 это одно и то же)

```
public Del_3
extern P1:word
.code

Del_3 proc
  push eax
  push ebx
  push edx
  xor  edx, edx
  mov  ax,P1
  mov  bx, 3
```

```
      imul bx
      cmp  dx, 0 ;если на dx что-то есть то вышли за диапазон слова
      jnz  Zero               1) НЕВЕРНО, dx=0FFFFh при P1<0 !!!
                                 Надо проверять ФЛАГИ !
      mov  P1, ax
      jmp outp
   Zero:
      mov P1, 0
outp:
      pop edx
      pop ebx
      pop eax
      ret
   Del_3 endp
   end
```

ОЦЕНКА=3

108-№6. Ибрагимова С.В.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

1. Вы не написали что такое X поэтому я приняла за x 108, за № 6
   1) X - ИМЯ ПЕРЕМЕННОЙ
   2) Так нельзя `X dw -2046; !!!!`

108 dw -2046
2046 это 11111111110
0000.0111.1111.1110
1111.1000.0000.0001
\+ 1

1111.1000.0000.0010 => F802 (как-то змейкой надо записать)

Ответ: 02F8

ОЦЕНКА=5

```
2. include                                                  console.inc
   .data
   x                                db                              ?
   .code
   Start:
         ClrScr
         inint                                                      x
         mov                         al,                           10
         cmp                          al,                          x
         JA                                                        E
         outstrln                                                "HET"
         JMP                                                       R
   E:    outstrln                                                "ДА"
   R:
         exit
         end Start
```

ОЦЕНКА=6

```
3. include                                                  console.inc
   .data
   x                                db                              ?
   .code
   Start:
```

```
        ClrScr
        mov                     eax,                    0
        mov                     ecx,                    0
   L:   inchar                                          x
        cmp                     x,                    "."
        JE                                             R
        cmp                     x,                    "&"
        JNE                                            T
        mov                     ecx,                    1
   T:   cmp                     x,                    "0"
        JB                                             E
        cmp                     x,                    "9"
        JA                                             E
        cmp                     ecx,                    0
        JE                                             E
   for                      i,                  <"2","0","4","6","8">
        local                                          Q
        mov                     ah,                    i
        cmp                     x,                    ah
        JNE                                            Q
        mov                     al,                   "0"
        sub                     x,                     al
        add dx, x           1)      НЕЛЬЗЯ,      dx=dw      <>      x=db
   Q:
   endm
   E:   JMP                                            L
   R:   outword dx          2)     Нет,     сумма     цифр     Longword
        exit
                            3) end Start
```

ОЦЕНКА=3

4.

```
   include                                                  console.inc
.data
N                                   equ                          10
A               db                  N               dup         (1)
Y                                    dw                           ?
.code
Start:
      ClrScr
      mov                               Y,                       0
      PUSH offset A             1)     НЕ     тот     ПОРЯДОК     ПАРАМЕТРОВ
      PUSH                                                       N
      PUSH                          offset                       Y
      call                                                   Summ
      outintln                                                   Y
      exit
Summ                                                         proc
      PUSH                                                    ebp
      mov                                   ebp,              esp
      PUSH                                                    eax
      PUSH                                                    ebx
      PUSH                                                    ecx
      PUSH                                                    edx
      PUSH                                                    esi
      PUSH                                                    edi
      mov                               eax,                     0
      mov                 ecx,                  [ebp+12];        N
      mov                esi,                   [ebp+16];       ^A
      mov             edi,                 [ebp+8];            ^Y
L:    mov ebx, [esi+ecx-1]        2)     Сразу     ЧЕТЫРЕ     элемента     !!!
      mov                           eax,                       ecx
```

```
      mul                                                              ebx
      add [edi], eax                3) Не было Y:=0
                                    4)      Y=dw      <>      eax=dd          !!!
      Loop                                                               L
      POP                                                              edi
      POP                                                              esi
      POP                                                              edx
      POP                                                              ecx
      POP                                                              ebx
      POP                                                              eax
      POP                                                              ebp
      RET                                                              4*3
Summ                                                                  endp
      end Start
```

ОЦЕНКА=2

5.

```
include console.inc
Sum_1 macro X:req
   local C, B, A           0) C=B=A Понять или спросить
if type X EQ 1             1) Не понято условие "только форматов m8, m16 или m32"
                              НЕ регистр
   mov bl, X
   xor eax, eax
   mov ecx, 8; 8 раз shr1 и сложим остатки
C:
   shl bl, 1
   adc al, 0; при shr остаток в CF
   Loop C                  2) Это число "1", а надо "0" !
elseif type X EQ 2
   mov bx, X
   xor eax, eax
   mov ecx, 16
B:
   shl bx, 1 ; вправо на 1
   adc al, 0
   Loop B
elseif type X EQ 4
   mov ebx, X
   xor eax, eax
   mov ecx, 32
A:
   shl ebx, 1
   adc al, 0
   Loop A
                               3) else ERROR ???
endif
endm
.data
X db 10101010b
.code
Start:
   Sum_1 X
   outu al
   exit
end Start
```

(я не уверенна простите за оформление, спешу)

ОЦЕНКА=1

108-№8. Какунин К.В.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

001	ВВЦ	100	200	000
002	МОД	019	100	020
003	ПЕР	302	000	018
004	ВЧЦ	000	302	019
005	ПБ	000	0	000
006	ВЧЦ	000	200	399
007	ПР	000	0	000
008	ПЕР	200	000	198
009	ПЕР	399	000	199
010	СЛЦ	198		

ОЦЕНКА=0

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
0 0 0 0  0 1 1 1  1 1 1 1  1 1 1 0
1 1 1 1  1 0 0 0  0 0 0 0  0 0 0 1
 +                                    1
1 1 1 1  1 0 0 0  0 0 0 0  0 0 1 0   <= Otvet: h208FØ

  F         2        0      8      1) НЕТ 02F8h
                                      Цифры в БАЙТЕ НЕ
                                      переставляются !!!
```

ОЦЕНКА=0

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

```
include console.inc
.DATA
X dd ?
.code
BEGIN:
    ININT X
    CMP X,100
    JB N
    CMP X,1000
    JAE N
    OUTSTR 'ДА'
    JMP F
N:
    OUTSTR 'НЕТ'
F:
    EXIT
    END BEGIN
```

ОЦЕНКА=6

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

```
INCLUDE settings.inc       1) console.inc (КАК С практикумом ???)
INCLUDE io2020.inc         2) Не надо
.STACK 4096
.DATA
   N=5
X DB N DUP (3)             3) НЕЛЬЗЯ, N не КОНСТАНТА !!!
Y DW ?
.CODE
P PROC
   PUSH EBP
   MOV  EBP,ESP
   PUSH ECX
   PUSH EBX
   PUSH EAX
   MOV word ptr [EBP+16],0       1) Не соответствует ВЫЗОВУ
                                 2) (АДРЕС Y):=0 ?????
   MOV ECX,[EBP+12]; N
   MOV BX,[EBP+16]
L: MOV AL,[EBP+8][ECX]-1         1) Не соответствует ВЫЗОВУ
   MUL ECX;AX=AL*ECX             3) Учит УМНОЖЕНИЕ
   ADD BX,AX
   LOOP L
   MOV word ptr[EBP]+8,BX        4)(АДРЕС Y):=bx ?????
                                 5) Не надо word ptr
   POP EAX
   POP EBX
   POP ECX
   POP EBP
   RET 4*3
P ENDP

start:
   PUSH offset X                 1) НЕ тот ПОРЯДОК !!!
   PUSH N
   PUSH offset Y
   CALL P
   EXIT
   END start
```

ОЦЕНКА=0

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

```
sum1 MACRO X
   LOCAL s,l,nxt
IF (opattr X NE 42)
   .err <Incorrect variable>     1) Учить type X
ELSE
   XOR EAX,EAX
s: CMP X,0;
   JE nxt
   SHR X                         2) ????
```

```
   JB l                          3) Надо проверить CF
   INC AL
l: JMP s
nxt:
ENDIF
OUTI EAX
ENDM
```

ОЦЕНКА=0

108-№7. Казакова Д.С.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*G-№); G – Ваша группа
```

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0 ; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

4. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

5. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_3. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем P1, описанной в каком-то другом модуле. Если значение Perem3 при этом выйдет за допустимый диапазон, то обнулить его.

Зачет; 26.05; Казакова Дарья 108 группа

Задание №1

Число: -(108 * 10 – 7) = -1073
Представление числа в двоичном виде:0000010000110001
Обратный код: 1111101111001110
Доп.код: 1111101111001111
В шестнадцатеричном виде(запись байтов в обратном порядке): CFFB
Ответ: CFFB
ОЦЕНКА=6

Задание №3

```
include console.inc
.data
.code
Start:
    xor ebx, ebx
w8:
    inchar al
    cmp al, '*'
    je main
    cmp al, '.'
    je fin
    jmp w8

main:
    inchar al
    cmp al,'.'
    je fin
    cmp al, '0'
    jb main
    cmp al, '9'
    ja main
    sub al, '0'
    test al, 1b
    jnz main
                                1) sub al,'0'
    movzx eax, al
    add ebx, eax
```

```
        jmp main
    fin:
       outword ebx,, The sum of digits that are multiples (7 mod 7 + 2) in the
line after" * " is equal to '       1) Нет ПЕРВОЙ '

        pause
        exit
        end Start
```

ОЦЕНКА=5

Задание №2

```
include console.inc
.data
T       dw ?

.code
Start:
        inint T
        mov ax, T
        cmp ax, 0
        jge skip
        neg ax                  1) А если нет такого ?
skip:
        xor ecx, ecx
        mov bx, 10
L:
        cmp ax, 0
        je fin                  2) Для X=0 надо ответ 1 !!!
        cwd                     3) НЕТ, уже БЕЗЗНАКОВОЕ !
                                   xor dx,dx
        div bx
        test dx, 1b
        jnZ L
        inc ecx
        jmp L

fin:
       outword ecx,, "The number of even numbers in the number: "

        pause
        exit
end Start
```

ОЦЕНКА=4

Задание №4

```
include console.inc
.data
n equ 700
MAS db n dup(n dup(?))
.code
DIAG proc
      push ebp
      mov  ebp, esp
      push ebx
      push ecx
      push edx
      push eax
```

```
    push esi

      mov ecx, [ebp+12]; N -> ecx
      mov edx, ecx
      mov ebx, [ebp+8]; OFFSET X -> ecx
      xor esi, esi
@STEP:
      mov al, [ebx+esi]
      cmp al, '0'
      jl  @NOT_NUM                        1) НЕТ jb @NOT_NUM; БЕЗЗНАКОВЫЕ
      cmp al, '9'
      jg  @NOT_NUM                        1) НЕТ ja
      mov al, 7
      mov [ebx+esi], al
@NOT_NUM:
      add esi, edx
      inc esi
      loop @STEP
for P, <esi, eax, edx, ecx, ebx, ebp>
      pop P
endm
      ret 2*4
DIAG endp
Start:

push n
push offset MAS
call DIAG


end Start
```

ОЦЕНКА=5

Задание №5

(Ошибка: переменная P1 заменена на Perem3)

```
include console.inc
.data
extern Perem3 : word
.code
public Del_3
Del_3 proc
    push eax
    push edx
    mov ax, 7; №
    imul Perem3
    cmp dx, 0                1) НЕВЕРНО! dx=-1 для P1<0
    jne L
    mov Perem3, ax
    jmp Next

  L:
    mov Perem3, 0

  Next:
    pop edx
    pop eax
    ret
Del_3 endp
end
```

ОЦЕНКА=5

108-№9. Клоков Т.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

3. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

4. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

5. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_3. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем P1, описанной в каком-то другом модуле. Если значение P1 при этом выйдет за допустимый диапазон, то обнулить его.

1)

101-1001 – элементы

```
001 ВВЦ 101 100 000; Вводим массив
002 МОД 000 101 012; x[i] mod 3
003 УСЛ 004 007 007; mod = 0
004 ВЧЦ 000 011 100; 0 - x[i]
005 УСЛ 008 007 008
006 СЛЦ 012 012 101; S = S + x[i]
007 СЛЦ 004 004 015; I + 1
   СЛЦ 002 002 015
   СЛЦ 006 006 016; не пронумеровал, т.к. нет времени сдвинуть коды
008 ВЧЦ 014 014 011
009 УСЛ 010 003 003; if k=0 тогда проходим дальше
010 ВЫЦ 012 001 000; write(s)
011 СТОП 000 000 000
012 00 000 000 003; const 3
013 00 000 000 000; S
014 00 000 000 100; k
015 00 000 001 000; 1
016 00 000 000 001
```

1) ??? Всего 512 !

2) Почему длина=100

3) НЕТ в 100 НИЧЕГО нет

4) 012=3 и СУММА !!!

ОЦЕНКА=0

---

2)

```
include console.inc
.data
      X db ?
.code
start:
      inint X
      mov ecx, 8                1) ?????
      xor ebx, ebx
Loop_1:
      movsx ax, X
      mov dl, 10
      div dl                    2) Надо ЗНАКОВОЕ деление по условию
      cmp ah, 0
      je  L1
      cmp ah, 2                 3) А если ah=-2 ???
      je  L1
      cmp ah, 4
      je  L1
      cmp ah, 6
```

```
       je  L1
       cmp ah, 8
       je  L1
       dec ecx                    4) В байте не 8, а всего ТРИ цифры !
Loop_end:
       cmp ecx, 0
       jne Loop_l
       jmp end_l
L1:
       add bl, ah
       jmp Loop_end
end_l:
       outword ebx
       exit
end start
```

ОЦЕНКА=0

---

3)

```
N eq 900                                  1) equ
.data
       Mas db N dup (db N dup (?))        2) НЕ надо db, учить
.code
F proc
       push ebp
       mov  ebp, esp
       push eax
       push ebx
       push ecx
       mov  cx, [ebp + 12]                3) Откуда процедура знает, что N=dw ?
       mov  ebx,[ebp + 8]
Loop_l:
       mov al, [ebx]
       cmp al, '0'
       jb  Loop_end
       cmp al, '9'
       ja  Loop_end
       mov byte ptr [ebx], 9
Loop_end:
       add ebx, [ebp + 12]                3) А здесь N=dd ???
       add ebx, 1
       dec cx
       cmp cx, 0                          4) Не надо, понять или спросить
       jne Loop_l
       pop ecx
       pop ebx
       pop eax
       pop ebp
       ret 8
F endp

start:
       push N
       push offset Mas
       call F
end start
```

ОЦЕНКА=3

---

4)

```
First_1 macro X:Req
       local Loop_l, L1
       push eax
       push ecx
       push ebx
if type X NE 1 and type X NE 2 and type X NE 4
       .err <Bad type>
       pop eax
       pop ecx
       pop ebx
       exitm
```

```
endif
;      mov al, type X
;      mov cl, 8
;      mul cl
       mov cx, ax                        1) ПРОСТО mov cx,(type X)*8
       xor ebx, ebx
Loop_l:
       shr X, 1
       jnc L1
       inc bl
L1:
       dec cx
       cmp cx, 0
       jne Loop_l
       mov cl, bl
if type X eq 1
       mov X, FFh
elseif type X eq 4                       2) eq 2
       mov X, FFFFh
else
       mov X, FFFFFFFFh
endif
       mov                               3) ????
       shl X, cl
       pop ebx
       pop ecx
       pop eax
endm
```

ОЦЕНКА=5

---

5)

```
.data
       extrn P1: dword     0) word
.code
public Del_3

Del_3 proc
       push eax
       push ebx
       push ecx
       push edx
       mov  ax, P1         1) ax=dw<>P1= dword !!!
       cwd                 2) Это УМНОЖЕНИЕ, а не ДЕЛЕНИЕ !!!
       mov  cx, 9
       imul cx
       cmp  dx, 0          3) НЕТ, dx=-1 для X<0 !!!
       je   Norma
       jne  Bolsh
Norma:
       mov P1, ax
       jmp end_l
Bolsh:
       mov P1, 0
End_l:
       pop edx
       pop ecx
       pop ebx
       pop eax
       ret
Del_3 endp

end
```

ОЦЕНКА=4

108-№10. Марченко Ф.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

5. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

108-№11. Муравицкая Е.Я.

1. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

№ = 11

```
         X  dw -(10*G-№); G – Ваша группа
-(10*108-11) = -1069
106910 = 2^10+2^5+2^3+2^2+2^0=00001100001011012
Обратный 1111101111010010
+1
1111101111010011 = FBD3
В машинном представлении: D3FB

ОЦЕНКА=6
```

---

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

№ = 11 =>11 mod 3 = 2

```
Include console.inc
.data
X dd ?
.code
Start
   Inint X
   Mov ecx, 0              0) sub ecx,ecx
   Mov ebx, 10
   Mov edx, 0
   Cmp X, 0
   jGE next1
   neg X                   1) Существует ???
next1:
   mov eax, X
L:
   Cmp eax, 0
   jE  fin                 2) НЕТ, для X=0 ответ=1 !
   test eax,1
   jNZ next2
   inc ecx
next2:
   div ebx
   mov edx,0
   jmp L
fin:
   outword ecx
   exit
end Start
```

ОЦЕНКА=5

---

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

№ = 11 =>11 mod 7 = 4 => 11 mod 7+2 = 6 => цифры, кратные 6

```
Include console.inc
.code
Start
   Mov ecx,0 ;sum
   mov ebx,6
L:
```

```
  Mov eax,0
  Inchar al
  Cmp al,'*'
  jE  fin
  cmp al,'0'                    1) Плохое управление:
                                   Здесь '*' ещё НЕ найдена !
  jb  L
  cmp al,'9'
  ja  L
  sub eax,'0' ;ord( c ) - ord(0)
  ;mov esi,eax
  ;mov edx,0
  ;div ebx
  ;cmp edx,0                    2) ПРОСТО cmp eax,6; !!!
  jNE  L
  add  ecx,esi
  jmp  L
fin:                            3) Не понято условие: не ДО
                                   первой '*', а ПОСЛЕ
  outword ecx
  exit
End Start
```

ОЦЕНКА=0

---

4. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа `MAS` и длину массива `N`. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ `chr(№)`. Привести пример вызова этой процедуры.

№ = 11 => n = 1100

```
P proc
    push ebp
    mov  ebp, esp
    push eax
    push ebx
    push ecx
    push edx
    mov ebx, [ebp + 8] ; offset X
    mov ecx, [ebp + 12] ; N
    mov edx, ecx;         N
    xor eax, eax ; eax = 0
L:
    mov al, [ebx]
    cmp al, '0'
    jb  next
    cmp al, '9'
    ja  next
    mov al, '+'                 1) Не понято условие задачи
                                   "на символ chr(№)" = 11
    mov [ebx], al
next:
    inc ebx
    add ebx, edx
    loop L

    pop edx
    pop ecx
    pop ebx
    pop eax
    pop ebp
    ret 8
```

```
P endp
-------------------------------------------------
    push n
    push offset X ; X – массив типа mas
    call P
```

ОЦЕНКА=2

---

5. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем `Del_3`. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем `P1`, описанной в каком-то другом модуле. Если значение `Perem3` при этом выйдет за допустимый диапазон, то обнулить его.

№ = 11

```
Include console.inc
Public Del_3
Extrn Perem3: word
.code
Del_3 proc
   Mov  ax, Perem3
   Mov  bx, 11
   Imul bx
   Mov  P1, ax
   Cmp  dx, 0              1) НЕТ dx=-1 для X<0 !!!
                              надо jo
   jNE next
   mov P1,0
next:
   ret
Del_3 endp
end
```

ОЦЕНКА=4

108-№12. Муравский Д.П.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

5. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

108-№13. Нестеров Д.А.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

2. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_3. Эта процедура должна умножать на № значение знаковой переменной размером в слово (**dw**) с именем P1, описанной в каком-то другом модуле. Если значение P1 при этом выйдет за допустимый диапазон, то обнулить его.

1 задача.

```
include console.inc
       .data
              X dw ? ; Nomer = 13 ==> 13 mod 3 = 1 ==> dw
       .code
Start:
       inint X
       mov  ax,X
       mov  cl, 1 ; for result
       mov  bx, 10
       cmp  ax, 0
       je   Out_cycle
       mov  cl, 0
Cycle:
       cmp  ax, 0
       je   Out_cycle
       cwd
       idiv bx
       shr  dx, 1
       jc   Next
       inc  cl
Next:
       jmp  Cycle
Out_cycle:
       outword cl
       EXIT
end Start
```

ОЦЕНКА=6

---

2 задача.

```
First_1 macro X:req
       local K,End_macro,Cycle,Next
       K=0
for i,<al,ah,bl,bh,cl,ch,dl,dh,ax,bx,cx,dx,si,di,bp,sp,eax,ebx \
        ecx,edx,esi,edi,esp,ebp>
  ifidni <i>,<X>
       K=1
       exitm
  endif
endm
       push eax
       push ebx
       push ecx
       push edx
if K EQ 1
       pop edx
       pop ecx
       pop ebx
       pop eax
       .err <Bad argument>
```

```
elseif type X EQ 4
	mov eax,X
	mov ecx,32
elseif type X EQ 2
	mov ax,X
	mov ecx,16
elseif type X EQ 1
	mov al,X
	mov ecx,8
else
	pop edx
	pop ecx
	pop ebx
	pop eax
	.err <Bad argument>
endif

	cmp X,0
	je End_macro
	mov bl,0 ; число единиц
Cycle:
	shr eax,1
	jnc Next
	inc bl
Next:
	loop Cycle
	mov X,0
if type X EQ 4
	mov X,080000000h ; X := 1000 0000 0000 0000 0000 0000 0000 0000b
elseif type X EQ 2
	mov X,08000h ; X := 1000 0000 0000 0000b
elseif type X EQ 1
	mov X, 080h ; X := 1000 0000b
endif

	mov cl, bl
	sar X,cl

End_macro:

	pop edx
	pop ecx
	pop ebx
	pop eax
	endm
```

ОЦЕНКА=6

---

3 задача.

; Nomer = 13

```
.data
	extern P1:word
.code
	public Del_3
Del_3 proc
; prologue
	push ebp
	mov  ebp, esp
	push eax
	push ebx
	push edx
	mov  ax, P1
	mov  bx, 13 ; N
	imul bx
	jo   L
	jmp  F
L:
	mov ax,0
```

```
                                        1) jmp !!!
F:
       mov P1,ax

; epilogue
       pop edx
       pop ebx
       pop eax
       pop ebp
       ret
Del_3 endp

end
```

ОЦЕНКА=5

108-№14. Почернина О.Л.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

2. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

3. Написать на Ассемблере неголовной модуль, содержащий описание процедуры без параметров с именем Del_5. Эта процедура должна делить на № значение беззнаковой переменной размером в слово (**dw**) с именем P2, описанной в каком-то другом модуле.

Задание 1.

```
include console.inc
.data
  X dd ?
.code
Start:
   inint X
   mov  ebx, 0             1) sub ebx,ebx
   mov  edi, 10
   mov  eax, X
L:
   test eax, eax
   jz   endd
   inc  ebx
   xor  edx, edx
   div  edi
   jmp  L
endd:
   cmp  ebx, 3
   je   da
   outstr "НЕТ"
   exit
da:
   outstr "ДА"
   exit
end Start
```

ОЦЕНКА=6

Задание 2.

```
Sum_1 macro X:req
        Local T
        T = 0
if type X eq 4
   for N, <eax,ebx,ecx,edx,edi,esi,ebp,esp>
     ifidni <X>,<N>
         .err <Bad argument(X)>
         exitm
     endif
   endm
   T = 32
elseif type X eq 2
   for N, <ax,bx,cx,dx,di,si,bp,sp>
     ifidni <X>,<N>
         .err <Bad argument(X)>
         exitm
     endif
```

```
   endm
         T = 16
elseif type X eq 1
   for N, <al,bl,cl,dl>
     ifidni <X>,<N>
         .err <Bad argument(X)>
         exitm
     endif
   endm
         T = 8
else
         .err <Bad argument(X)>
         exitm
endif
         xor eax, eax
repeat T
         Local Skip
         rol X, 1
         jc Skip; if CF = 1 => symbol was 1 => doesn't count
         inc EAX
Skip:
endm
         endm
```

ОЦЕНКА=6

Задание 3.

```
include console.inc
   EXTERN P2: word
.code
                            1) public Del_5
Del_5 proc
   push ecx
   push eax
   push edx
   mov  ax, P2
   xor  dx, dx
   mov  cx, 14
   div  cx
   mov  P2, ax ;; P2 := ax div 14
   pop  edx
   pop  eax
   pop  ecx
   ret  0
Del_5 endp
   end
```

ОЦЕНКА=4

108-№16. Сахаутдинова А.Р.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2, расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

5. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

Сахаутдинова Аделина 108, первая итерация

№2.

X dw – 2046

2046 = 0000011111111110

-2046 =11111000 00000010 $_2$ = F8 02 $_{16}$

В двоичном виде

X

00000010
11111000

В шестнадцатеричном виде

X

02
F8

ОЦЕНКА=6

№3.

```
include console.inc
.data
X dw ?
.code
Start:
   inint X
   cmp X, 100
   jae N
   cmp X, 10
   jb N
   outstrln “ДА”
   jmp F
N: outstrln “НЕТ”
F: pause
   exit
   end Start
```

ОЦЕНКА=6

№4.

```
include console.inc
.data
```

```
.code
Start:
   mov ebx, 0              1) sub
L1:
   inchar al
   cmp al, '.'
   je F  ;  если символ '&' так и не был встречен
   cmp al, '&'
   jne L1
L2:
   inchar al
   cmp al, '.'
   je F
   cmp al, '6'
   jne L2
   add ebx, 6
   jmp L2
F:
   outword ebx
   pause
   exit
   end Start
```

ОЦЕНКА=6

№5.

```
Include console.inc
.data
N equ 300
A db N dup (1)
Y dw ?
.code
ArrSum proc
   push ebp
   mov  ebp, esp
   push eax
   push ebx
   push ecx
   push edx
   push esi
   mov ebx, [ebp + 8]; ^A
   mov esi, [ebp + 16]; ^Y
   mov ecx, 0                                1) sub
@L:
   movzx eax, byte ptr [ebx][ecx]
   outintln eax, , 'eax = '                  2) ?????
   inc ecx
   mul ecx
   add [esi] , ax
   cmp ecx, [ebp + 12]
   jne @L
   pop esi
   pop edx
   pop ecx
   pop ebx
   pop eax
   pop ebp
   ret 4*3
ArrSum endp
Start:
   push offset Y
   push N
```

```
pish offset A
call ArrSum
outwordln Y
pause
exit
end Start
```

ОЦЕНКА=6

№1

№	ком	A1	A2	A3	Комментарий
001					
002					
003					
004					
005					
005					
006					
007					
008					
009					
010					
011					
012					
013					
014					
015					
016					

ОЦЕНКА=0

108-№15. Савицкий И.П.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*G-№); G – Ваша группа
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

4. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

ЗАДАНИЕ 3

```
include console.inc
.data
   X db ?
.code
Start:
   inint X
   mov ecx, 3
   xor ebx, ebx
   mov dl, 100; поскольку максимум 3 цифры — идем со старшего разряда чтобы учесть
незначящие нули
   mov al, X
L:
   mov ah, 0                  1) Не понято условие "целое знаковое число X"
   div dl                     1) Не понято условие "целое знаковое число X"
   test al, 1
   jnz cont
   cmp al, 0
   jne sure
   test bh, 1 ; на bh флаг значящего/незначящего нуля
   jz cont
sure:
   inc bl ; на bl - ответ
   mov bh, 1
cont:
   mov al, ah
   loop L
   outint bl                  2) Не понято условие " по outword"
end Start
```

ОЦЕНКА=0

ЗАДАНИЕ 1

<не дорешено, можно не смотреть>
001 ввц 019 015 000
002 вчц 000 017 018
003 пмр 000 000 000 переход по <0
004 мод 000 017 012 остаток от деления
005 пнр 000 000 000 переход по <> 0
006 слц 016 016 013 s = s + 1
008 пр 000 000 000 переход по = 0
009 слц 017 017 013 i = i + 1

010 без 000 000 000
011 выв 016 001 000 вывод s
012 стоп 000 000 000 остановка
013 000 000 001
014 000 000 003
015 000 108 000 — количество чисел
016 000 000 000 s
017 000 000 019 i
018 000 000 000 — const 0
019 <начало X>

ОЦЕНКА=0

ЗАДАНИЕ 2

```
-(10*G-№) = -(10 * 108 — 15) = -1065

1065_10 = 10000101001_2 (при помощи _n я указываю систему счисления)
0000 0100 0010 1001(потому что dw)
1111 1011 1101 0110
0000 0000 0000 0001
1111 1011 1101 0111 — финальное представление в памяти     1) Наоборот !
```

ОЦЕНКА=3

ЗАДАНИЕ 4

```
diag proc
       push ebp
       mov  ebp, esp
       push edx
       push ecx
       push eax
       push ebx
       mov  edx, [ebp + 8] ;адрес начала матрицы
       mov  ecx, [ebp + 12] ;количество строк/столбов
       mov  ebx, ecx ;сохраняем это количество где-то где оно не изменялось
L:
       mov  al, byte ptr [edx] ; берем очередной символ           1) Не надо byte ptr
       cmp  al, '0' ; проверка на символ
       jb   continue
       cmp  al, '9'
       ja   continue
       mov  byte ptr [edx], 15 ; ставим наш символ если проверку прошли
continue:
       add  edx, ebx ; в этих двух строках мы переходим на следующий элемент диагонали
       inc  edx          ;
       loop L
       pop ebx
       pop eax
       pop ecx
       pop edx
       pop ebp
       ret 2*4
diag endp

Start:
       push N ; тут размер
       push offset X ; тут массив
       call diag
       exit
end Start
```

ОЦЕНКА=6

ЗАДАНИЕ 5

```
First_1 macro X:Req
```

```
    local reg_flag, L, continue
    reg_flag = 1 ; 0 - регистр
for reg, <al,ah,bl,bh,cl,ch,dl,dh, \
          ax,bx,cx,dx,si,di,       \
          eax,ebx,ecx,edx,esi,edi>
  ifidni <reg>,<X>
    reg_flag = 0
  endif
    exitm                                1) Выше, в ifidni
                                            Выход, дальше обработки НЕТ !
if reg_flag EQ byte
    push eax
    push ecx
    mov eax, 0
  if type X EQ 1
    mov ecx, 8
L:
    shr x, 1                             2) БОЛЬШОЕ X !!
    jnc continue
    or al, 10000000b
    shr al, 1
continue:
    loop L
    mov X, al
elseif type X EQ word
    mov ecx, 16
L:
    shr X, 1
    jnc continue
    or ax, 1000000000000000b
    shr ax, 1
continue:
    loop L
    mov X, ax
elseif type X EQ dword
    mov ecx, 32
L:
    shr X, 1
    jnc continue
    or eax, 10000000000000000000000000000000b
    shr eax, 1
continue:
    loop L
    mov X, eax
endif; закрыли проверку на размер
    pop ecx
    pop eax
else
    .err <неверный формат аргумента>
endif

     endm
```

ОЦЕНКА=3

108-№21. Шамков И.В.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

4. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

№1

Некорректное условие в памяти УМ-3 всего лишь 512 ячеек, как бы мне не хотелось, но помещать туда 100*108 я не смогу, задача нерешаемая

ОЦЕНКА=?

№4.

```
include console.inc
.data
   N equ 100                  1) НЕТ, N equ 2100
   X db N dup (?)             2) НЕТ, матрица
.code
P proc
   push ebp
   mov ebp, esp
   push eax
   push ebx
   push ecx
   push edx; edx будет указывать на диагональные элементы
   push edi;
   mov ebx, [ebp + 8]; сюда адрес начала массива X
   mov ecx, [ebp + 12]; сюда N
L: mov byte ptr [ebx], '#'          3) Не ТА задача
   inc edi                          4) edi НИЧЕМУ не равно !
   add ebx, ecx
   add ebx, 1
   Loop L; тоесть мы ровно n раз перейдём на диагональный элемент
   finish:
         pop edi
         pop edx
         pop ecx
         pop ebx
         pop eax
         pop ebp
         ret 2*4
P endp
Start:
   push N
   push offset X
   call P
   exit
end Start
```

ОЦЕНКА=0

№3.

```
include console.inc
.data
N   equ 21; мой номер в списке
.code
Start:
; пусть сумма в esi
; в bl храним наше N mod 7 + 2
   mov al, N
   div al,7; не пров. на переполнение  1) Плохое деление ah=?
   add al, 2
   mov bl, al                          2) ПРОСТО mov bl,2
   xor edi, edi
; пусть в edi будет счётчик *
Rep_inp:
   inint al
   cmp al, '.'
   je Finish
   cmp al,'*'
   jne than
   mov edi, 1
than:
   cmp al, '0'
   jb  not_sum
   cmp al, '9'
   ja  not_sum
   cbw; расширили al до ax             3) НЕТ, символ - беззнаковый
   div bl; остаток в ah
   cmp ah, 0
   jne not_sum
   cwde; расширяем ax до eax           4) НЕТ, символ - беззнаковый
   cmp edi, 1
   jne not_sum
   add esi, eax
not_sum:
   jmp Rep_inp
finish:
   outword esi
   exit
end Start
```

ОЦЕНКА=2

№2

```
include console.inc
.data
; N=21
; значит ввожу db
   X db ?
.code
Start:
   inint X; значащие цифры до тех пор пока x <>0 после div 10

; пусть счётчик цифр в ecx                    1) Где ecx:=0 ???
   movsx ax, X; так как al делить нельзя
L: div 10                                     2) Учить ДЕЛЕНИЕ
   cmp al, 0
   je end_digit                               3) А цифра в ah ?
   shr ah, 1; делим цифру на 2 остаток на CF
   jc not_sum; если остаток 1
   inc ecx
```

```
not_sum:
   cbw; то есть мы вернули на ax результат деления ax := word(al)
   jmp  L
end_digit:
   outword ecx
   exit
end Start
```

ОЦЕНКА=0

№5

```
include console.inc
First_1 macro X:req
   local L1,L11, L2, L22, L33, Lend
```

1) Где L3 ???
2) L1=L2=L3 и L11=L22=L33
   Понять или спросить

```
if type X EQ 1
   mov  al, X
   mov  ecx, 8
   xor  ebx, ebx; счётчик 1
   xor  edi, edi; счётчик 0
L1:shr  al, 1; остаток на CF
   adc  ebx, 0; суммируем все 1
   Loop L1
   mov  edi, 8
   sub  edi, ebx
   mov  al, X
   mov ecx, ebx; в n количество единиц
   cmp  ecx, 0
   je   Lend
```

3) Lend НЕВИДИМА
   стоит в elseif type X EQ 4

```
   mov  dl, 10000000b; маска
   mov al, dl; тоесть по сути присвоили
L11:
   shr dl,1
   or al, dl; сдвинутое на одну позицию 1
   Loop L11
   mov X, al
; аналогично только word
elseif type X EQ 2
   mov ax, X
   mov ecx, 16
   xor ebx, ebx; счётчик 1
   xor edi, edi; счётчик 0
L2:shr ax, 1; остаток на CF
   adc ebx, 0; суммируем все 1
   Loop L2
   mov edi, 16
   sub edi, ebx
   mov ax, X
   mov ecx, ebx; в n количество единиц
   cmp ecx, 0
   je Lend
```

3) Lend НЕВИДИМА
   стоит в elseif type X EQ 4

```
   mov dx, 1000000000000000b; маска
   mov ax, dx; то есть по сути ax := dx
L22:
   shr dx,1
   or ax, dx; сдвинутое на одну позицию 1 логически складываем
   Loop L22
   mov X, ax
elseif type X EQ 4
```

```
   mov eax, X
   mov ecx, 32
   xor ebx, ebx; счётчик 1
   xor edi, edi; счётчик 0
L3: shr eax, 1; остаток на CF
   adc ebx, 0; суммируем все 1
   Loop L2
   mov edi, 32
   sub edi, ebx
   mov eax, X
   mov ecx, ebx; в n количество единиц
   cmp ecx, 0
   je Lend                                        3) Lend НЕВИДИМА
                                                     стоит в elseif type X EQ 4
   mov edx, 80000000h; маска
   mov eax, edx; то есть по сути ax := dx
L33:
   shr edx,1
   or eax, edx; сдвинутое на одну позицию 1 логически складываем
   Loop L33
   mov X, eax
Lend: mov X, 0                                    4) То есть ВСЕГДА X:=0 !!!
; то есть число равно нулю
   endif
endm
```

ОЦЕНКА=3

108-№18. Скворцова А.Ю.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
Sum_1 macro X:Req
```

Оно должно записывать в регистр AL количество "0" (битов со значением ноль) во внутреннем машинном представлении параметра X. Параметр X может быть только форматов m8, m16 или m32, например, для X **db** 10101010b должно получаться AL=4. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

№1:

```
01 ВВЦ   100  200  000 ; Read(x)
02 МОД   000  100  013 ; 000:= x[i] mod 2 ; i := 100, j := 299 в начале
03 УСЛ   011  004  004 ; Если mod := 0, то печатаем 1 и на выход
04 ВЧЦ   000  100  299 ; 000:= x[i] - x[j]
05 УСЛ   006  011  011 ; Если x[i] <> x[j], то прыгаем на выход
06 СЛЦ   004  004  017 ; изменяем i и j
07 ВЧЦ   016  016  015 ;  S := S - 1
08 УСЛ   009  004  004 ; (да, меньше 0 у нас никогда не будет, но так красивее выглядит)
09 ВЫЦ   014  001  000 ; выводим 0
10 БЕЗ   000  012  000
11 ВЫЦ   015  001  000 ; выводим 1
12 СТОП  000 000  000
13 00    000 000  002 ; const 2
14 00    000 000  000 ; const 0
15 00    000 000  001 ; const 1
16 00    000 000  100 ; S :=100
17 00    000 000  511 ; хитрая дельта
```

ОЦЕНКА=6

№2:

1) 0000.0111.1111.1110
2) 1111.1000.0000.0001
3) +1
4) 1111.1000.0000.0010, т.е. F802. Тогда в машине это будет 02F8 – ответ

ОЦЕНКА=6

№3:

```
include console.inc
```

```
.data
   X db (?) ; 1 байт
.code
begin:
   mov ebx, 0 ; счетчик          1) sub
   inint X
   movzx eax, X
   mov   ecx, 10
L:mov edx, 0; для деления
   inc bx
   div ecx
   cmp eax,0
   ja  L
   cmp ebx, 1
   je V1
   outstr "No"
   jmp KON
V1:outstr "Yes"
KON:
   exit
end begin
```

ОЦЕНКА=6

№4:

```
push offset Y
push N
push offset X
call P

P proc
   push ebp
   mov  ebp, esp
   pushad
   mov edi, [ebp+8] ; offset x
   mov ebx, [ebp+12] ; N
   mov ecx, [ebp+16] ; offset Y
   mov edx, 0                    1) sub
   mov [ecx], edx ; Y := 0       2) НЕТ, Y=dw <> edx=dd
L: mov esi, [ebp+12];
   dec esi, ebx                  3) ?????
   inc esi ; esi := i            3) ?????
   mov eax, [edi]                4) Сразу 4 элемента массива !
   mul esi
   add [ecx], eax                2) НЕТ, Y=dw <> edx=dd
   inc edi
   dec ebx
   cmp ebx, 0
   ja    L
   popad
   pop ebp
                                 5) ret !!!!
P endp
```

ОЦЕНКА=0

№5:

```
Sum_1 macro X
   local L1, K1, L2, K2, L3, K3       1) L1=L2=L3; K1=K2=K3
                                         Понято или спросить
```

```
   mov eax, 0
if type x EQ 1
    mov cl, x
L1:
    shr cl, 1
    jnc K1
    add al, 1
K1:
    cmp cl, 0
    ja L1
elseif type x EQ 2
    mov cx, x
L2:
    shr cx, 1
    jnc K2
    add al, 1
K2:
    cmp cx, 0
    ja L2
elseif type x EQ 4
    mov ecx, x
L3:
    shr ecx, 1
    jnc K3
    add al, 1
K3:
    cmp ecx, 0
    ja L3
else
   .err <Bad argument!>
   exitm
endif
   endm
```

2) Не понято условие задачи "только форматов m8, m16 или m32" Регистры - НЕЛЬЗЯ

ОЦЕНКА=2

108-№19. Толеутаева А.Б. Зачёт 26.05. Первая итерация

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

№	коп	A1	A2	A3	Комментарий
001	ВЦВ	200	800	000	Массив в 200:999 1) > 511 !
002	ВЧЦ	000	017	018	<>0 ?
003	ПБР	000	007	000	Неотрицательное число - выход
004	МОД	000	017	012	Кратность 3 2) НЕТ работы с X[i]
005	ПНР	000	007	000	Остаток <> 0 - выход
006	СЛЦ	016	016	011	S := S + 1
007	ВЧЦ	000	015	017	I – 999 = 0?
008	ПР	000	013	000	
009	СЛЦ	017	017	011	I := I + 1
010	БЕЗ	000	002	000	
011		000	000	000	Const = 1
012		000	000	000	Const = 3
013	ВЫВ	016	001	000	
014	СТОП	000	000	000	
015		000	000	999	Const = 999
016		000	000	000	S
017		000	000	200	i 2) 800 ???
018		000	000	000	Const = 0

ОЦЕНКА=0 НЕТ РАБОТЫ С массивом, учить !

---

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

N mod 3 = 1

```
.686
include console.inc

.data
T dw ?
.code
Start:
   Inint T ;целое знаковое
   cmp T, 0
   jg  next1
   neg T
next1:
   xor ebx, ebx ;answer
   mov ecx, 10
   mov eax, dword ptr T              1) НЕТ, movzx eax,T    !!!
next:
   div ecx ;edx - Остаток            2) Плохое деление, edx=?
   shr edx, 1
   jc  L
```

```
   inc ebx ;чёт
L:
   cmp eax, 0
   ja  next
   outword ebx
   exit
   end Start
```

ОЦЕНКА=2

---

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword. N mod 7 + 2 = 5+2 = 7. Кратны только 0 и 7.

```
include console.inc
.data
flag db 0
.code
Start:
   sub ebx, ebx ;sum
cycle:
   inchar cl
   cmp cl, '.'
   je  epilogue
   cmp cl, '*'
   je  L
   cmp flag, 0
   je cycle
L: mov flag, 1
   cmp cl, '0'
   je  adding
   cmp cl, '7'
   je  adding
   jmp cycle
adding:
   inc ebx                1) Не понято условие задачи "сумму цифр"
   jmp cycle

epilogue:
   outword ebx
   exit
   end Start
```

ОЦЕНКА=3

---

4. Пусть на Паскале дано описание типа массива:

```
const n=100*19 = 1900; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(19). Привести пример вызова этой процедуры.

```
P proc
   Push ebp
   Mov  ebp, esp
   Pusha                                0) pushad
   mov edx, [ebp + 8]; адрес Mas[1,1]
   mov ecx, [ebp + 12]; N, ecx - счётчик
   mov esi, ecx ;N
L:
   Mov edi, dword ptr [edx]             1) НЕТ, это 4 символа !!
   Cmp edi, '0'
   Jb  next
   Cmp edi, '9'
   Ja  next
   Mov byte ptr [edx], 19
```

```
Next:
   Add edx, esi
   Inc edx
   Loop L
   Popa
   Pop ebp
   Ret 2*4
P endp
```

Вызов

```
Start:
   push N
   push offset Mas
   call P
   exit
```

ОЦЕНКА=2 (Процедура) 0 (массив)

---

5. Написать макроопределение с заголовком

```
First_1 macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов m8, m16 или m32, ставить в этом параметре в начало все "1" (биты со значением единица) во внутреннем машинном представлении X, например, для X **db** 10101010b необходимо получить X **db** 11110000b. Макроопределение должно настраиваться на тип параметра и выдавать необходимую диагностику о неверном типе своего операнда.

```
First_1 macro X:Req
      local K, CYCLE, CYCLE1, CYCLE2, next, next1, next2

               1) CYCLE = CYCLE1= CYCLE2 - Понять или спросить
               1) next = next1= next2 - Понять или спросить

      K = 0 ;  0 - не найдено ошибки

for reg, <al,ah,bl,bh,cl,ch,dl,dh, /
          ax,bx,cx,dx,si,di,bp,sp, /
          eax,ebx,ecx,edx,esi,edi,ebp,esp>
  ifidni <reg>,<X>
      K = 1
  endif
      exitm                      2) Выше, в ifidni
                                    Выход, дальше обработки НЕТ !

if K EQ 1
     .err <Macros arg can not be register>

                                 3) exitm  !!!

elseif type X EQ 1
      push EAX
      push ECX
      mov  AL, 0
      mov  ECX, 8
CYCLE1:
      shr X, 1
      jnc next1
      or  AL, 10000000b
      shr AL, 1
next1:
      loop CYCLE1
      mov  X, AL
      pop  ECX
      pop  EAX

elseif type X EQ 2
```

```
     push EAX
     push ECX
     mov AX, 0
     mov ECX, 8                  4) НЕТ, 16
CYCLE2:
     shr X, 1
     jnc next2
     or  AX, 1000000000000000b
     shr AX, 1
next2:
     loop CYCLE2
     mov X, AX
     pop ECX
     pop EAX
elseif type X EQ 4
     push EAX
     push ECX
     mov EAX, 0
     mov ECX, 8                  5) НЕТ, 32
CYCLE:
     shr X, 1
     jnc next
     or  EAX, 10000000000000000b 6) НЕТ, надо 32 бита !
     shr EAX, 1
next:
     loop CYCLE
     mov X, EAX
     pop ECX
     pop EAX
else
     .err <Macros arg can be only m8,m16,m32>
endif
      endm
```

ОЦЕНКА=2

108-№20. Тунис И.С.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200, затем печатать целое число 0, если введённый массив X начинается с нечётного числа и симметричен (одинаково читается слева направо и справа налево), в противном случае программа печатает целое число 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X dw -2046
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое беззнаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outstr**) слово "ДА", если в десятичной записи введённое число содержит № **mod** 3+1 количество значащих цифр (т.е. таких цифр, удаление которых меняет величину числа), иначе выводит слово "НЕТ".

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 6+2расположенных в этом тексте после первого "&". Считать, что эта сумма не более MaxLongword.

5. Написать *процедуру* на Ассемблере, которая получает в качестве параметров адрес начала массива A целых беззнаковых чисел в формате *байта*, длину этого массива N≥300 (массив считается пронумерованным от 1 до N) и адрес целочисленной переменной Y в формате слова. Процедура возвращает в Y сумму 1*A[1]+2*A[2]+... N*A[N]. Переполнение результата игнорировать. Процедура обязана выполнять стандартные соглашения о связях. Привести пример вызова этой процедуры.

№1

ОЦЕНКА=0

№2

2046 = 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0

1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1

+ 1

1 1 1 1 1 0 0 0 0 0 0 0 0 0 1 0

Answer: h208FØ

1) НЕТ 02F8h
   Цифры в БАЙТЕ НЕ
   переставляются !!!

ОЦЕНКА=0

№3

```
include console.inc
.data X dd ?
.code
Begin:
   inint X
   cmp   X,100
   jb    nott
   cmp   X,999
   ja    nott
   outstr 'DA'
   jmp   fin
nott:
   outstr 'NET'
fin:
   exit
   end Begin
```

ОЦЕНКА=6

№4

```
include console.inc
.data
```

```
.code
Begin:
   mov bl,1
   mov eax, 0              1) sub eax,eax
   mov ecx, 0              1) sub ecx,ecx
cikl:
   inchar al
   cmp al, '.'
   je  fin
   cmp bl,1
   jne nextt
   cmp al, '&'
   jne eend
   mov bl,0
   jmp eend
nextt:
   cmp al,'0'
   je  yess
   cmp al, '4'
   je  yess
   cmp al, '8'
   je  yess
   jmp eend
yess:
   sub al, '0'
   add ecx, eax
eend:
   jmp cikl
fin:
   outword ecx
   exit
   end Begin
```

ОЦЕНКА=6

№5

```
Sum proc
   push EBP
   mov EBP, ESP
   pushad
   mov ESI, [EBP + 8] ; offset A
   mov ECX, [EBP + 12]; N
   mov EDI, [EBP + 16]; offset Y
   mov EBX, 1
@L:
   movzx EAX, byte ptr [ESI]
   mul EBX
   add [EDI], AX
   inc EBX
   inc ESI
   cmp EBX, ECX
   jbe @L
   popad
   pop EBP
   ret 4*3
```

ОЦЕНКА=6

108-№5. Жуков Н.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 100*G (G – номер Вашей группы), затем печатать число S, равное количеству одновременно отрицательных и кратных трём элементов массива X. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X (в двоичном или шестнадцатеричном виде):

```
X  dw -(10*G-№); G – Ваша группа
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **T** (T=**db** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**dd** для № **mod** 3=2). Программа выводит (по **outword**) число значащих *чётных* цифр (т.е. '0', '2', '4', '6', '8') в десятичной записи значения числа X. Цифра является значащей, если её удаление меняет величину числа.

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму цифр, кратных № **mod** 7+2, которые расположены в этом тексте после первой "*". Считать, что таких цифр не более MaxLongword.

5. Пусть на Паскале дано описание типа массива:

```
const n=100*№; type MAS=array[1..n,1..n] of char;
```

Написать на Ассемблере *процедуру* со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS и длину массива N. Процедура должна заменить все символы *цифр* на главной диагонали этой матрицы на символ chr(№). Привести пример вызова этой процедуры.

№1 НЕТ

№2

-(10*G-№)=-1075

1075=0000010000110011

1111101111001100

1111101111001101

1111101111001101=FBCD

ОТВЕТ ØFBCDh 1) Наоборот, перевёрнутое

ОЦЕНКА=3

№3

```
   INCLUDE                                                SETTINGS.INC
INCLUDE                                                     IO2020.INC
.DATA
X                                  DD                                ?
.CODE
START:
      ININT                                                        X
      MOV                                                      EAX,X
      XOR                                                    EDX,EDX
      MOV                                                     EBX,10
      XOR                                                      CX,CX
      CMP                                                      EAX,0
      JE                                                         Zer
      JG                                                         Cyc
      NEG                                                        EAX
Cyc:                               DIV                             EBX
      SHR                                                      EDX,1
      JC                                                         Cnt
      INC                                                         CX
      XOR EDX,EDX                1)   Это   надо   после   Cnt:      !!!
Cnt:                                CMP                        EAX,0
      JNE                                                        Cyc
      JMP                                                        FIN
Zer:                                MOV                       CX,1
      JMP                                                        FIN
FIN:                               OUTWORD                          CX
      NEWLINE
```

```
      EXIT
END START
```
ОЦЕНКА=4

№4

```
;5 mod 7+2 = 7
include console.inc
.data
   X DB ?

.code
Start:
     XOR ECX,ECX
Rd:
     INCHAR AL
     MOV X[ECX],AL         1) Нельзя! X НЕ массив, ОДИН СИМВОЛ
     INC ECX
     CMP AL,'.'
     JNE Rd
     DEC ECX
     MOV AL,'*'
     MOV EDI,offset X           2) Задачу с массивом НЕ РЕШИТЬ
     CLD
     REPNE SCASB
     JNE FIN
     XOR AX,AX
Fnd:
     CMP byte ptr [EDI][ECX-1],'7' ;единственная цифра, кратная 7
     JNE Cnt
     ADD AX,7
Cnt:
     LOOP Fnd
     OUTU EAX
FIN:
end Start
```
ОЦЕНКА=0

№5

```
rt proc:                                          1) УЧИТЬ ОПИСАНИЕ процедуры
    push    ebp
    mov     ebp,esp
    push    ebx
    push    esi
    mov     ebx,dword ptr [ebp+8];X
    mov     ax,word ptr [ebp+12]; Y               2) Это НЕ ТА ЗАДАЧА
            N equ 500
            mov     cx,1
            cmp     ax,cx
            jl      @L1
            dec     cx
@L3:
            inc     cx
            mov     dx,cx
            movzx   edx,dx
            imul    esi,edx,N
            mov     dx,cx
            movzx   edx,dx
            lea     esi,dword ptr [ebx+esi-N-1]
            cmp     byte ptr [esi+edx],'0'
```

```
            jnae    @L2
            cmp     byte ptr [esi+edx],'9'
            jnbe    @L2
            mov     byte ptr [esi+edx],5
    @L2:
            cmp     ax,cx
            jg      @L3
    @L1:
            pop     esi
            pop     ebx
            leave
            ret     8
    rt endp
```

Вызов

```
PUSH                                                                    n
PUSH                                    offset                          M
CALL rt
```

ОЦЕНКА=0

107-№2. Балабин Д.С.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **dd**, и выводит ближайшее к X целое число, которое больше X и кратно №. Если нужное число больше MaxLongInt, то вывести "Большое число". Например, для X=-18 и №=12 надо вывести ответ -12.

```
;№ = 2
include console.inc
.data
   X dd ?
.code
Start:
   inint X
   mov eax,X
   mov bx,10
   xor ecx,ecx
   xor edx,edx
L:
   cmp eax,0
   jg P
N:
   test eax,1
   jne S
   add eax,2
   outint eax
   jmp Kon
P:
   test eax,1
   jne S
   add eax,2
   jo Per
   outint eax
   jmp Kon
S:
   add eax,1
   jo Per
   outint eax
   jmp Kon
Per:
   outstr 'Большое число'
Kon:
   end Start
```

ОЦЕНКА=6

---

2. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 0, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.

```
; № = 2
include console.inc
.data
   ch1 db ?
.code
Start:
   xor edx,edx
   xor ebx,ebx
   xor eax,eax
L:
   inchar ch1
   movzx ebx, ch1
   cmp   ebx, '.'
   je    ED1
   cmp   ebx, 2
   je    R
   cmp   ebx, '0'
   jb    L
   cmp   ebx, '9'
   ja    L
   sub   ebx,'0'
```

```
   test  ebx,1
   je    L
   add   eax, ebx
   Jmp   L
R:
   mov   edx,1
ED1:
   cmp   edx,1
   jne   ED2
   outword eax
   Jmp   ED
ED2:
   Outword 0
ED:
end Start
```

ОЦЕНКА=6

---

3. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

```
P proc
   push ebp
   mov  ebp,esp
   push eax
   push ebx
   push ecx
   push edx
   push edi
   mov  ebx,[ebp+8]; x
   mov  word ptr ecx,[ebp+12]; n     1) Нет формата word ptr ecx
                                     2) n=dd > 2 байта !

   mov  dx, word ptr [ebx+ecx*2-4];предпоследний элемент
                                     3) Не надо word ptr
   xor  edi,edi
L:
   mov  ax,word ptr [ebx]            3) Не надо word ptr
   cmp  ax,dx
   jge  R
   cwde
   add  edi,eax
R:
   add  ebx,2
   Loop L
   mov dword ptr [ebp+16],edi        4) Не надо dword ptr
                                     5) Это (Адрес Y):=edi;
                                        а надо Y:=edi
                                        НЕТ ОТВЕТА процедуры !
   pop edi
   pop edx
   pop ecx
   pop ebx
   pop eax
   pop ebp
   ret 12
P endp

   Push  offset Y
   movzx eax,N
   Push  eax
   push  offset x
   call P
```

ОЦЕНКА=2

---

4. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

```
include console.inc
ProBit macro x:req
   local L,P,ED
if type x EQ 4
   mov ecx,32                          0) Плохо, если x=ecx
elseif type x EQ 2
   mov ecx,16
elseif type x EQ 1
   mov ecx,8                           1) Просто mov ecx,8*(type x) !
else
   echo 'Bad argument x'
   .err
   exitm
endif
   xor edx,edx                         0) Плохо, если x=edx
   xor eax,eax                         0) Плохо, если x=eax
L:
   shr x,1
   jc  P
   add dl,1
   jmp ED
P:
   add al,1
ED:
   Loop L
   mul  dl
if type x EQ 4
   movzx eax,ax
   mov   x,eax
elseif type x EQ 2
   mov   x,ax
elseif type x EQ 1
   mov x,al
endif
   endm

.data
.code
start:
exit
end start
```

ОЦЕНКА=4

---

5. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

```
mov ax,A
and ax,mask y          1) mask Y
shr ax,y               2) Y
outword ax
```

ОЦЕНКА=5

107-№3. Бобровская А.С.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.
2. Выписать вид внутреннего машинного представления целой переменной X

```
X dw -(15*G+1-№); G - номер Вашей группы
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.
4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 1, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.
5. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G - номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

107-№8. Кожеуров П.С.

1. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

2. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

3. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

1)

```
Sum_min proc
    push    ebp
    mov     ebp, esp
    push    eax
    push    ebx
    push    ecx
    push    edx
    push    esi
    sub    eax, eax
    mov    esi, [ebp+8];  ^X
    mov    ecx, [ebp+12];  N
    mov    bx, [esi+ecx*2-4]
@@:
    cmp    bx, [esi]
    jle    L
    movsx  edx, bx
    add    eax, edx
L:
    add    esi, 2
    loop   @B
    mov    esi, [ebp+16]
    mov    [esi], eax
    pop    esi
    pop    edx
    pop    ecx
    pop    ebx
    pop    eax
    pop    ebp
    ret    12
Sum_min endp

      push offset Y
;     mov  eax, N
;     push eax                              1) push N
      push offset A
      call Sum_min
```

ОЦЕНКА=6

---

2)

```
          Nomer record X:2,Y:№,Z:7

      mov   eax, A
      ;and   eax, mask Y  необязательно т.к. Y занимает ровно al
      shr   eax, Y
      outword    al ;я не до конца понял в каком
; исчислении выводить поле поэтому
```

```
; далее я вывожу в двоичном                1) Это в 10-м виде !
      mov   ecx, 8
@@:
      sub   dx, dx
      shl   al, 1
      adc   dx, 0
      outword   dx
loop  @B
```

ОЦЕНКА=6

---

3)

```
data
  extern _F@0: near     1) F@0
  Param dw ?
.code
      public Param
Start:
      call _F@0               2) F@0
                          3) Что сделано с результатом F ?
      exit
      end  Start
```

ОЦЕНКА=4

107-№10. Козуб Д.В.

1. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 0, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.

В задаче написано выводить сумму. Не понял, причем тут произведение. Очепатка

```
include console.inc
.code
start:
sub ecx, ecx
sub ebx, ebx
L:
inchar al
cmp  al, '0'
jb   Next_L
cmp  al, '9'
ja   Next_L
mov  bl, al
sub  bl, '0'
test bl, 1
jz   Next_L
add  ecx, ebx
Next_L:
cmp  al, 10                1) Почему ЦИФРА не может иметь
                                 НОМЕР 10 в алфавите ?
je   Est_simvol
cmp  al, '.'
jne  L
;sub  dx, dx
;outword dx                2) outword 0 ?
jmp  Finish
Est_simvol:
outword ecx
Finish:
    exit
    end start
```

ОЦЕНКА=5

---

2. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

```
F proc
push ebp
mov  epb, esp
push eax
push ebx
push ecx
push edx
push edi
push esi
mov  ebx, [ebp + 8] ; var X
mov  ecx, [ebp + 12]       ; N
mov  eax, [ebp + 16]       ; var Y
mov  di, [ebx + 2*ecx – 4] ; предпоследний
sub  esi, esi       ; кол-во
L:
cmp  [ebx], di
jge  Next_L
movsx edx, word ptr [ebx]
add  esi, edx
Next_L:
add  ebx, 2
loop L
```

```
mov [eax], esi
pop esi
pop edi
pop edx
pop ecx
pop ebx
pop eax
pop ebp
ret 12
F endp
```

Вызов:

```
N equ 214000
.data
X MAS dup N (?)
Y dd ?
.code
push offset Y
mov  ebx, N
push ebx                    1) Просто push N
push offset X
call F
```

ОЦЕНКА=6

---

3. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

```
ProBit macro X:Req
       local K, L
       K = type X
if K EQ 1
       mov  cl, X
       sub  al, al          ; кол-во нулей
L:
       shl  cl, 1
       adc  al, 0
       test cl, cl
       jnz  L
       mov  cl, al
       neg  cl
       add  cl, 8           ; кол-во единиц
       mul  cl
       mov  X, al
elseif K EQ 2
       mov  cx, X
       sub  ax, ax   ; кол-во нулей
L:
       shl  cx, 1
       adc ax, 0
       test cx, cx
       jnz L
       mov cx, ax
       neg cx
       add cx, 16    ; кол-во единиц
       mul cx
       mov X, ax
elseif  K EQ 4
       mov ecx, X
       sub eax, eax  ; кол-во нулей
L:
       shl ecx, 1
       adc eax, 0
       test ecx, ecx
```

```
       jnz L
       mov ecx, eax
       neg ecx
       add ecx, 32   ; кол-во единиц
       mul ecx
       mov X, eax
else
       echo Wrong argument
       .err
       exitm
endif
       endm                                1) Ну, очень много команд
```

ОЦЕНКА=6-

---

4. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw)** с именем Param, описанной в головном модуле.

В задаче не сказано описывать другой модуль, кроме головного

```
include console.inc
.data
public Param
Param dw 10
.code
start:
sub    eax, eax
extrn  F@0: near
call   F@0
outint eax
    exit
end start
```

ОЦЕНКА=6

107-№12. Москалев А.Р.

1. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 0, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.

```
include console.inc
.code
start:
       sub eax, eax
       sub ebx, ebx
       sub ecx, ecx
       sub edx, edx
Cycle:
       inchar al
       cmp al, 12
       jne Next
       mov cl, 1
Next:
       cmp cl, 0
       jne No_sum
       cmp al, '0'
       jb No_sum
       cmp al, '9'
       ja No_sum
       mov bl, al
       and bl, 1           1) Почему чётные цифры на чётных номерах алфавита ?
       cmp bl, 1           2) Не надо, уже есть флаги
       jne No_sum
       add edx, eax        3) Нет, надо sub al,'0' !
No_sum:
       cmp al, '.'
       jne Cycle
       cmp cl, 0
       jne Outword         4) Нельзя, имя макро !
       sub edx, edx
Outword:                   4) Нельзя, имя макро !
       outword edx
       exit
       end start
```

ОЦЕНКА=3

---

2. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

```
P proc
  push ebp
  mov  ebp, esp
  push eax
  push ebx
  push ecx
  push edx
  push edi
  push esi
  mov  ebx, [ebp + 8]; ^X
  mov  ecx, [ebp + 12]; N
; mov  eax, ecx
; sub  eax, 2
; sub  edx, edx                  1) Зачем ? Это НЕ деление !
; mov  edi, 2
; mul  edi
; add ebx, eax
  movsx edx, word ptr [ebx]      2) Просто
                                    movsx edx, word ptr [ebx+2*ecx-4]
                                    УЧИТЬ !
```

```
; mov ebx, [ebp + 8]
  sub esi, esi
L:
  movsx ax, word ptr [ebx]        3) НЕЛЬЗЯ 2 байта РАСШИРИТЬ до 2-х байт
  cmp eax, edx
  jge Next
  add esi, eax
Next:
  add ebx, 2
  Loop L
  mov eax, [ebp + 16]
  mov [eax], esi
  pop esi
  pop edi
  pop edx
  pop ecx
  pop ebx
  pop eax
  pop ebp
  ret 12
P endp

push offset Y
push N
push offset X
call P
```

ОЦЕНКА=4

---

3. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

```
ProBit macro X:Req
      local L
if type X EQ 1
      mov bl, X
      xor al, al
      mov cl, 8
L:    shl bl, 1
      adc al, 0
      dec cl
      jnz L
      mov bl, 8
      sub bl, al
      mul bl
      mov X, al
elseif type X EQ 2
      mov bx, X
      xor ax, ax
      mov cl, 16
L:    shl bx, 1
      adc ax, 0
      dec cl
      jnz L
      mov bx, 16
      sub bx, ax
      mul bx
      mov X, ax
elseif type X EQ 4
      mov ebx, X
      xor eax, eax
      mov cl, 32
L:    shl ebx, 1
      adc eax, 0
      dec cl
```

```
        jnz L
        mov ebx, 32
        sub ebx, eax
        mul ebx
        mov X, eax
else
        .err <Wrong argument in macros ProBit>
endif
        endm
```

ОЦЕНКА=6

---

4. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw)** с именем Param, описанной в головном модуле.

```
public Param
extern F@0:near
.data
Param dw 12
.code
Start:
        call F@0
                              1) Где печать результата функции ?
        exit
        end Start
```

ОЦЕНКА=4

107-№13. Мурин Е.А.

1. Выписать вид внутреннего машинного представления целой переменной X
```
X dw -(15*G+1-№); G – номер Вашей группы

0C7F9h
```

ОЦЕНКА=6

---

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для №**mod** 3=0; T=**dw** для №**mod** 3=1; T=**db** для №**mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.

```
include console.inc
.data
      X dw ?
.code
start:
      ININT X
      mov  ax, X
      mov  ecx, 0                    1) sub
      mov  ebx, 0                    1) sub
circle:
      cmp  ax, 0
      je   endofprog
      inc  ecx                       2) Не понято условие
                                        " самый правый бит числа имеет номер 0"
      test ax, 1
      jz   del
      add  ebx, ecx
del:
      shr ax, 1
      jmp circle
endofprog:
      OUTWORD ebx
      exit
      end start
```

ОЦЕНКА=3

---

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 1, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.

```
include console.inc
.code
start:
      xor eax, eax
      xor ecx, ecx
circle:
      INCHAR al
      cmp  al, 13
      je   read_all
      cmp  al, '.'
      je   no_13
      cmp  al, '0'
      jb   circle
      cmp  al, '9'
```

```
      ja   circle
      sub  al, '0'
      test al, 1
      jnz circle
      add ecx, eax
      jmp circle
read_all:
      INCHAR al
      cmp al, '.'
      jne read_all
      jmp endofp
no_13:
      mov ecx, 1
endofp:
      OUTWORD ecx
exit
end start
```

ОЦЕНКА=6

---

4. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G – номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

```
.data
      X dw 107000 dup (?)
      Y dd ?
.code
      Public P                              1) Зачем ?
P proc
      push ebp
      mov  ebp, esp
      push eax
      push ebx
      push ecx
      push edx
      push edi
      mov  ecx, [ebp + 12]; N
      mov  ebx, [ebp + 8]; ebx - ссылка на первый эл
;     sub  ecx, 2
      mov  dx, [ebx + ecx * 2]            2) [ebx + ecx * 2-4]
;     add  ecx, 2
      xor  eax, eax
circle :
      cmp [ebx], dx
      jbe  next
      movzx edi, word ptr [ebx]
      add  eax, edi
next :
      add  ebx, 2
      loop circle
      mov  edx, [ebp + 16]
      mov  [edx], eax
      pop  edi
```

```
      pop  edx
      pop  ecx
      pop  ebx
      pop  eax
      pop  ebp
      ret  3 * 4
P endp
start :
      mov ecx, 15
full :
      mov X[ecx], cx
      loop full
      push offset Y
      push 15
      push offset X
      call P
end
```

ОЦЕНКА=6

---

5. Написать макроопределение с заголовком

```
RazBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `r16`, `m8`, `m16`, `r32` или `m32`, печатать (по **`outint`**) разность количества бит со значением "1" и количеством бит со значением "0" во внутреннем машинном представлении параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики об ошибках обращения и неверных типах своего операнда.

```
   RazBit macro X : Req
if type X EQ 0                1) А если type X=3 bkb <0 ???
      .err
      exitm
endif
If type X NE 4
      movzx eax, X
else
      mov  eax, X
endif
      mov  ecx, 0
circle :  ; считаем количество «1»
      cmp  eax, 0
      je   endofm
      test eax, 1
      jz   del
      inc  ecx
del :
      shr  eax, 1
      jmp  circle
endofm :
      mov eax, (type x)*8
;     mov ebx, 8
;     mul ebx ; eax = количество нулей
      sub ecx, eax ; разность
      OUTWORD ecx
endm
```

ОЦЕНКА=5

107-№16. Петров П.В.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.

ОЦЕНКА=0

---

2. Выписать вид внутреннего машинного представления целой переменной X в 16-м виде

X **dw** -(15*G+2-№); G – номер Вашей группы

-(15*107+2-16)=-1591=1111 1001 1100 1001=F9C9 (В памяти: 1001001110011111=9C9F)

1) НЕТ 0C9F9h
   В памяти переставляются БАЙТЫ, а не ЦИФРЫ в байте

ОЦЕНКА=2

---

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **dd**, и выводит ближайшее к X целое число, которое больше X и кратно №. Если нужное число больше MaxLongInt, то вывести "Большое число". Например, для X=-18 и №=12 надо вывести ответ -12.

```
include console.inc
.const
msg db 'Большое число'
.data
X dd ?
.code
start:
    inint X
    mov  eax, X
    cdq
    mov  bl, 16
    idiv bl                    1) Учить ДЕЛЕНИЕ
    imul eax, bl               2) Учить УМНОЖЕНИЕ
    cmp edx, 0
    jb next                    3) НЕТ, знаковые !
    add eax, 16                4) МОЖЕТ быть > MaxLongInt
next:
    cmp eax, MaxLongInt        5) MaxLongInt в Ассемблере НЕТ
    jne chislo
    outword msg
    jmp fin
chislo:
    outword eax
fin:
    exit
```

ОЦЕНКА=0

---

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 0, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.

```
include console.inc
.const
zerr db '0'
.code
start:
    xor ecx, ecx
nachalo:
    inchar al
    cmp  al, '0'          1) Сначала надо проверять al=chr(№)
                             вдруг chr(№) - цифра ?
    jb   next
    cmp  al, '9'
    ja   next
    mov  bl, al
```

```
        add  bl, '0'        2) sub ???
        push bl             3) НЕЛЬЗЯ !
        test bl, 1
        jz  next
        pop bl              3) НЕЛЬЗЯ !
        add ecx, bl         4) НЕЛЬЗЯ !
    next:
        cmp al, '.'
        je  zero
        cmp al, chr(16)     5) Нет такого в Ассемблере
        je  vivod
        jmp start           6) jmp nachalo ???
    vivod:
        outword ecx
        jmp fin
    zero:
        outword zerr
    fin:
        exit
        end start
```

ОЦЕНКА=0

---

5. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

```
P   proc
    push ebp
    mov  ebp, esp
    push eax
    push ebx
    push esi
    push edi
    mov ecx, [ebp + 12]; N
    mov esi, [ebp + 8]; ^X
    add esi, ecx
    sub esi, 2
    mov eax, [esi] ; eax - предпоследний элемент
                                          1) НЕТ, длина X[i]=2 байта
                                          2) Это ПОСЛЕДНИЙ
    mov esi, [ebp + 8]
    xor ebx, ebx ; S:=0
cycle:
    mov edx, [esi + ecx - 1]        1) НЕТ, длина X[i]=2 байта
                                          3) edx:=Сразу 2 элемента
    cmp edx, eax
    ja  next                              4) НЕТ, знаковые
    add ebx, edx
next:
    loop cycle
    mov esi, [ebp + 16]
    mov dword ptr[esi], ebx         5) Не надо dword ptr
    pop edi
    pop esi
    pop ebx
    pop eax
    mov esp, ebp
    pop ebp
    ret 12
P   endp
```

6) Где пример вызова ?

ОЦЕНКА=0

107-№17. Семенов П.А.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.
2. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G – номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.
3. Написать макроопределение с заголовком

```
RazBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов r8, r16, m8, m16, r32 или m32, печатать (по **outint**) разность количества бит со значением "1" и количеством бит со значением "0" во внутреннем машинном представлении параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики об ошибках обращения и неверных типах своего операнда.
4. Дано описание

```
Nomer record A:3,B:№,C:4
X Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле B переменной X.
5. Написать на Ассемблере головной модуль, который вызывает процедуру без параметров P, описанную в каком-то другом модуле. Эта процедура должна делить на № значение беззнаковой переменную размером в слово (**dw**) с именем Perem, описанную в головном модуле.

1.

```
include console.inc

.data
      T db ?
.code
start:
      inint T
      mov al, T
      sub ecx, ecx
      sub edx, edx
loop_start:
      mov bl, 1
      and bl, al
      shr al, 1
      cmp bl, 1                1) Учить команду test
      je  if_1
continue_loop:
      add ecx, 1
      cmp al, 0
      jne loop_start
      outword edx
      exit
if_1:
      add edx, ecx
      jmp continue_loop
      end start
```

ОЦЕНКА=6-

2.

```
F proc
      push ebp
      mov  ebp, esp
      push eax
      mov  eax, [ebp+8]; var MAS
      push ecx
      mov  ecx, [ebp+12]; N
      push edx
```

```
      mov  dx, [ebp+16]; Y                          1) НЕТ, (Адрес Y)=dd
      push ebx
      sub  ebx, ebx
      sub  ecx, 1
start_loop:
      cmp word ptr [eax + 1], word ptr [eax]        2) Нельзя память-памть
                                                    3) НЕ предпоследний
                                                    3) НЕ x[i]
      ja  if_greater
loop_continue:
      add eax, 1                                    4) НЕТ add eax,2
      sub ecx, 1
      cmp ecx, 0                                    5) НЕ надо, есть флаги
      jne start_loop
      lmp end_loop                                  6) lmp ???
if_greater:
      mov bx, word ptr [eax+1]                      7) НЕ НАДО word ptr
                                                    3) НЕ x[i]
      add dword ptr [dx], ebx                       8) НЕТ ФОРМАТА
      jmp loop_continue

end_loop:
      pop dx                                        9) НЕЛЬЗЯ !!!
      pop ecx
      pop eax
                                                    10) Где ret ???
F endp

      push offset Y
;     mov  eax, N
;     push eax                1) push N
      push offset A
      call D                  2) Не то имя
```

ОЦЕНКА=0

3.

```
RazBit macro X:req
            local L1, L2, L3        0) L4 ?
if type X EQ 0                  1) А если type X=3 или <0 ?
      .err <Bad argument>
                                    2) exitm
endif
if type X EQ 1
      push al                       3) НЕЛЬЗЯ !
      push cl                       3) НЕЛЬЗЯ !
      push bl                       3) НЕЛЬЗЯ !
      mov  al, X
      sub  cl, cl
L1:
      mov bl, 1                     4) Учить test
      and bl, al
      cmp bl, 1
      je  L2
L3:
      shr al, 1
      cmp al, 0
      jne L1
      jmp L4
L2:
      add cl, 1
      jmp L3
L4:
      add cl, cl
```

```
      sub cl, 8
      outint cl
      pop bl
      pop cl
      pop al
endif
if type X EQ 2
      push ax                          3) НЕЛЬЗЯ !
      push cx                          3) НЕЛЬЗЯ !
      push bx                          3) НЕЛЬЗЯ !
      mov ax, X
      sub cx, cx
L1:
      mov bx, 1
      and bx, ax
      cmp bx, 1
      je  L2
L3:
      shr ax, 1
      cmp ax, 0
      jne L1
      jmp L4
L2:
      add cx, 1
      jmp L3
L4:
      add cx, cx
      sub cx, 16
      outint cx
      pop bx                           3) НЕЛЬЗЯ !
      pop cx                           3) НЕЛЬЗЯ !
      pop ax                           3) НЕЛЬЗЯ !
endif
if type X EQ 4
      push eax
      push ecx
      push ebx
      mov eax, X
      sub ecx, ecx
L1:
      mov ebx, 1
      and ebx, eax
      cmp ebx, 1
      je  L2
L3:
      shr eax, 1
      cmp eax, 0
      jne L1
      jmp L4
L2:
      add ecx, 1
      jmp L3
L4:
      add ecx, ecx
      sub ecx, 16
      outint ecx
      pop ebx
      pop ecx
      pop eax
endif
      endm
```

ОЦЕНКА=2

4.

```
      mov eax, X
      and eax, mask B
      shr eax, B
      outword eax
```

ОЦЕНКА=6

5.

```
.data
      extern P: near
      Perem dw ?

.code
start:
      public Perem
      call P
      exit
end start
```

ОЦЕНКА=6

107-№20. Сычев П.С.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.
2. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 0, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.
3. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.
4. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.
5. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

№2

```
include console.inc
.code
start:
      sub ecx, ecx
      sub ebx, ebx
      sub eax, eax
      sub edx, edx
O:
      inchar bl
      cmp bl, '.'
      je  konec
      mov bh, bl
      sub bh, '0'
      cmp bh, 20              1) НЕТ, надо cmp bl,20 !!!
```

```
      jne P
      add edx, ecx
      jmp konec
P:
      cmp bl, '0'
      jb  L
      cmp bl, '9'
      ja  L
      mov bh, bl
      sub bh, '0'
      test bh, 1
      jne  N
      jmp  L
N:
      movzx eax, bh
      add ecx, eax
L:
      jmp O
konec:
      outword edx
      exit
      end start
```

ОЦЕНКА=5

№3

```
Mass proc
    push ebp
    mov  ebp,esp;
    push eax
    push ebx
    push ecx
    push edx
    push esi
    mov  ecx,[ebp+12]; N
    mov  ebx,[ebp+8]; ^X
    mov  eax, dword ptr [ebx+2*ecx-4]
    mov  edx, [ebp+16]
    mov  esi, 0
    ;add ebx, 2 ;perviy element
L:
    cmp eax, dword ptr[ebx]
    jae No
    add esi, dword ptr [ebx]
No:
    add  ebx,2
    loop L ;цикл
    mov [edx], esi
    pop esi
    pop edx
    pop ecx ;возвращаем
    pop ebx
    pop eax
    pop ebp
    ret 12
Mass endp
```

1) Не надо dword ptr
2) eax:=сразу 2 э-нта !

3)sub

4) ???

5) Нет сравнения dd<>dw !
6) НЕТ, знаковые
7) сразу 2 э-нта !

ВЫЗОВ:

```
push offset Y
push 214000
push offset Mas
call Mass
```

ОЦЕНКА=0

№5

```
.data
public Param
Param dw ?
.code
start:

;F proc
  ;push eax
  ;push ebx
                                1) extrn Param: word
 ; mov ax, Param
 ; mov bx, 20                   так как она в другом модуле
  ;imul bx                            можно не описывать
 ; mov Param, ax          2) Не понято условие, F - функция !
 ; pop ebx
  ;pop eax
  ;ret
;F endp

extrn F@0 : near
call F@0
                                3) Ф-ция возвращает результат на eax !
exit
end start
```

ОЦЕНКА=2

№1

```
1  ввц      100   220   000
2  вчц      000   100   015
3  усл      010   004   004
4  мод      000   100   014
5  усл      006   007   007
6  слц      012   012   100; S=S+[i]
7  слц      002   002   013; i=i+1        1) НЕТ !
                                          2) Нет корр. адреса в
                                             ячейках 004 и 006
8  вчц      016   016   013; n=n-1
9  пнр      000   002   002               3) пнр 000 002 000
10 выц      012   001   000; S
11 стоп     000   000   000
12 00 000   000   000; S
13 00 000   000   001;1
14 00 000   000   002;2
15 00 000   000   000;0
16 00 000   000   220 ;n
```

ОЦЕНКА=2

№4

ОЦЕНКА=0

107-№21. Трофимов А.Д.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.
2. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 1, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.
3. Дано описание

```
Nomer record A:3,B:№,C:4
X Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле B переменной X.
4. Написать на Ассемблере головной модуль, который вызывает процедуру без параметров P, описанную в каком-то другом модуле. Эта процедура должна делить на № значение беззнаковой переменную размером в слово (**dw**) с именем Perem, описанную в головном модуле.

Задача 2.

```
Include console.inc
.data
      Sum dd (0)
.code
Start:
      Mov ebx, 1; ebx:=1 (сюда положим в конце сумму, если будет элемент 21)
      Xor eax, eax; eax:=0
L:    Inchar al
      Cmp al, '.'
      Jz  Fin
      Cmp al, 21
      Jz  E
      Cmp al, '0'             1) Зачем ??
      Jz  A
      Cmp al, '2'
      Jz  A
      Cmp al, '4'
      Jz  A
      Cmp al, '6'
      Jz  A
      Cmp al, '8'
;     Jz  A                   2) jnz L
;     Jmp L
A:    sub al, '0'
      Add Sum, eax
      Jmp L
E:    inchar al
      Cmp al, '.'
      Jnz E
      Mov ebx, Sum
Fin:  outword ebx
      Exit
      End Start
```

ОЦЕНКА=6

Задача 4.

```
include console.inc
.data
      public Perem
      Perem dw ?
.code
start:
      extrn P@0: near
```

```
      call P@0
      exit
end start
```

; Описание процедуры

```
      public P
P proc
      push ebp
      mov  ebp, esp
      push eax
      push ebx
      push edx
      movzx eax, Perem
      sub edx, edx
      mov ebx, 21
      div ebx
      mov Perem, ax
      pop edx
      pop ebx
      pop eax
      pop ebp
      ret
      P endp
                              1) end
```

ОЦЕНКА=5

Задача 3.

```
Nomer record A:3, B:21, C:4
X Nomer <>
...

      Mov eax, X
      And eax, mask B;    eax = <0,B,0>
      Shr eax, B;         eax = <0,0,B>
      Outword eax
...
```

ОЦЕНКА=6

Задача 1

№ **div** 2=21 **div** 2=10-й столбец

Точки, потому что excel не хотел вводить числа, начинающиеся с нуля

1	ВВЦ	.100	.225	.000	
2	СЛЦ	.007	.007	.100	
3	СЛЦ	.002	.002	.008	
4	ВЧЦ	.009	.009	.009	
5	ПБ	.000	.002	.000	
6	ВЫЦ	.007	.001	.000	
7	.00	.000	.000	.000	s = 0
8	.00	.000	.000	.015	const 15
9	.00	.000	.000	.015	n
10	.00	.000	.000	.001	const 1

1) Не понято условие задачи "в столбце матрицы с номером № **div** 2"

ОЦЕНКА=0

108-№2. Батушин А.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.

3. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

3. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов r8, m8, r16, m16, r32 или m32, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

4. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

5. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

Задача 4 (№=2)

```
Nomer record X:2,Y:№,Z:7
A Nomer <>

      mov ax, A
      and ax, mask Y
      shr ax, Y
      outint ax               1) outword ax
```

ОЦЕНКА=5

Задача 5

```
include console.inc
.data
      public Param
Param dw ?
.code
start:
      extrn F@0 : near
      call F@0
                              0) Что сделано срезультатом ф-ции
                              0) Не понято условие задачи
      exit
      end start
```

На всякий случай написал вспомогательный модуль с функцией F

```
include console.inc
.data
extrn Param : word
.code
public F
F proc
      push eax          1) НЕЛЬЗЯ, функция !
      push ebx
      mov ax, Param
```

```
      mov bx, 2; №
      imul bx
      mov Param, ax
      pop ebx
      pop eax           2) Не понято условие "функцию без параметров F"
F endp
      end
```

ОЦЕНКА=2

Задача 2:

```
include console.inc
.data
A dw 1,2,10,5,1
Y dd ?
.code
N EQU 5
P proc
      push ebp
      mov ebp, esp
      pushad
      mov eax, [ebp + 8]; ^X
      mov ebx, [ebp + 12]; N
      mov ecx, ebx
      mov bx, [eax + 2*ebx - 4]; предпоследний эл-т

      mov esi, 0                    1) sub
@L:
      cmp [eax],bx
      jge SKIP
      movsx edx, word ptr [eax]
      add esi, edx
SKIP:
      add eax, 2
      dec ecx
      cmp ecx, 1                    2) Не надо, уже есть флаги
      jge @L                        3) НЕТ, N - беззнаковое !
      mov edx, [ebp + 16]
      mov [edx], esi
      popad
      pop ebp
      ret 4*3
P endp

Start:
      push offset Y
      push N
      push offset A
      call P
      outint Y
      pause
      exit
end Start
```

ОЦЕНКА=5

Задача 3:

```
ProBit macro X: req

if type X EQ qword
      echo err
                        1) .err     Понять
                        2) exitm    Понять
```

```
endif
if type X EQ 0          3) А если type X=0,3,5,<0 ???
      echo err
                        1) .err     Понять
                        2) exitm    Понять
endif
      push eax
      mov ax, 0
repeat (type X) * 8
      local ZERO, ONE
      shr  X, 1
      jc   ONE
      inc  al
      jmp  ZERO
ONE:
      inc  ah
ZERO:
endm
      mul  ah
      push eax
      mov  eax, [esp + 4]
      mov  X, [esp]
      add esp, 8
endm
```

ОЦЕНКА=3

Задача 1:

```
001 ВВЦ 100 200 000           1) Не понято условие "длины 200+№"
002 ВЧЦ 000 100 020
003 ПР  000 012 000
004 МОД 000 100 019
005 ПНР 000 007 000
006 СЛЦ 017 100 017
007 СЛЦ 002 002 022
008 СЛЦ 004 004 022
009 СЛЦ 006 006 022
010 ВЧЦ 000 002 021
011 ПНР 000 002 000
012 ВЧЦ 000 017 020
013 ПНР 000 015 000
014 СЛЦ 017 017 018
015 ВЫЦ 017 001 000
016 СТОП 000 000 000
017 00  000 000 000; Сумма - 0
018 00  000 000 001; 1 - константа
019 00  000 000 002; 2 - константа
020 00  000 000 000; 0 - константа
021 ВЧЦ 000 299 018                       2) ВЧЦ 000 299 020
                                             см. Тунис
022 00  000 001 000
```

ОЦЕНКА=0

108-№2. Батушин А.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.

3. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

3. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

4. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

5. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

Задача 4

```
Nomer record X:2,Y:№,Z:7
A Nomer <>

…
mov ax, A
and ax, mask Y
shr ax, Y
outint ax
…
```

Задача 5

```
include console.inc

.data
public Param
Param dw ?

.code
start:
 ...
extrn F@0 : near
call F@0

exit
end start
```

На всякий случай написал вспомогательный модуль с функцией F

```
include console.inc

.data
extrn Param : word

.code
public F
F proc
push eax
push ebx

mov ax, Param
mov bx, 2; №
imul bx

mov Param, ax

pop ebx
pop eax
F endp

end
```

Задача 2:

```
include console.inc
.data
A dw 1,2,10,5,1
Y dd ?
.code

N EQU 5

P proc
push ebp
mov ebp, esp

pushad
mov eax, [ebp + 8]
mov ebx, [ebp + 12]
mov ecx, ebx

mov bx, [eax + 2*ebx - 4]; предпоследний эл-т
mov esi, 0

@L:
cmp [eax], bx
jge SKIP
movsx edx, word ptr [eax]
add esi, edx

SKIP:
add eax, 2
dec ecx
cmp ecx, 1
jge @L

mov edx, [ebp + 16]
mov [edx], esi

popad
pop ebp
ret 4*3

P endp

Start:
push offset Y
push N
push offset A
call P
outint Y

pause
exit
end Start
```

Задача 3:

```
ProBit macro X: req

if type X EQ qword
echo err
endif
if type X EQ 0
echo err
endif

push eax
mov ax, 0
repeat (type X) * 8

local ZERO, ONE
shr X, 1
jc ONE
inc al
jmp ZERO
```

```
ONE:
inc ah

ZERO:
endm
mul ah

push eax
mov eax, [esp + 4]
mov X, [esp]
add esp, 8

endm
```

Задача 1:

```
001 ВВЦ 100 200 000
002 ВЧЦ 000 100 020
003 ПР 000 012 000
004 МОД 000 100 019
005 ПНР 000 007 000
006 СЛЦ 017 100 017
007 СЛЦ 002 002 022
008 СЛЦ 004 004 022
009 СЛЦ 006 006 022
010 ВЧЦ 000 002 021
011 ПНР 000 002 000
012 ВЧЦ 000 017 020
013 ПНР 000 015 000
014 СЛЦ 017 017 018
015 ВЫЦ 017 001 000
016 СТОП 000 000 000
017 00 000 000 000          Сумма - 0
018 00 000 000 001          1 - константа
019 00 000 000 002          2 - константа
020 00 000 000 000          0 - константа
021 ВЧЦ 000 299 018
022 00 000 001 000
```

108-№4. Денисов В.М.

1. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **dd**, и выводит ближайшее к X целое число, которое больше X и кратно №. Если нужное число больше MaxLongInt, то вывести "Большое число". Например, для X=-18 и №=12 надо вывести ответ -12.

```
include console.inc
.data?
X dd ?
.code
Start:
    inint eax
    mov  X, eax
    xor  edx, edx                1) НЕТ, знаковое !
    mov  ebx, 4
    idiv ebx
    mov ebx, 4
    sub ebx, edx
    mov eax,X
    add eax,edx
    jno L
    outstrln "Большое число"
L:
    mov eax, X
    add eax, ebx
    outint eax
                                        2) exit
end Start
```

ОЦЕНКА=4

---

2. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

```
summ proc
      push ebp
      mov  ebp,esp
      push edx
      push ecx
      push ebx
      push eax
      mov  ebx, [ebp+8] ;offset MAS       0) MAS - ТИП !
      mov  ecx, [ebp+12] ;N
      mov  edx, [ebx + (ecx-2)*4 ]; предпоследний элемент
                                          1) НЕТ, X[i]=dw !
                                          2) edx:=Сразу 2 элемента !
      eax,  eax; for Y                    3) ????
@@:
      cmp [ebx], edx                      2) edx:=Сразу 2 элемента !
      jge bolshe
      add eax, [ebx]                      2) Сразу 2 элемента !
bolshe:
      add ebx, 4                          4) НЕТ add ebx,2
      loop @B
      mov [ebp+16], eax ;Y = sum          5) НЕТ, это (АДРЕС Y):=eax
                                             а надо Y:=eax
      pop eax
      pop ebx
      pop ecx
      pop edx
      pop ebp
```

```
      ret 3*4
summ endp
```

вызов

```
      push Y                                  6) НЕТ, var Y
      push N
      push MAS                                6) НЕТ, var
```

ОЦЕНКА=0

---

3. Написать макроопределение с заголовком

```
            ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

```
.data
ProBit macro X:req
    local L,  One, En
if type X EQ 1
  ifindi <X>, <bh>
        mov ah, bh
        movsx ebx, ah          1) Зачем movsx ?
  endif
        movsx ebx, X                 2) НЕЛЬЗЯ, ebx=dd <> X=db
        mov ch, 8
                                     3) endif ???
if type X EQ 2
  ifindi <X>, <bx>
        mov ax, bx
        movsx ebx, ax
  endif
                                     3) endif ???
        movsx ebx, X                 2) НЕЛЬЗЯ, ebx=dd <> X=dw
        mov ch, 16
else
  if type X EQ 4
        mov ebx, X
        mov ch, 32
    else
        .err <Bad arguments: constants>
        exitm
    endif
                                     4) Нарушена структура if !
    xor al, al
    xor ah, ah
    xor cl, cl
    mov edx, 1
L:
    inc cl
    cmp cl, ch
    je  n
    shl edx, 1
    test ebx, edx              5) УЧИТЬ работу с CF !
    jnz One
    inc al
    jmp L
One:
    inc ah
    jmp L
En:
    mul ah
if type X EQ 1
    mov X, al
```

```
else if type X EQ 2                  6) elseif ???
    mov X, ax
else if type X EQ 4                  6) elseif ???
    movsx ebx, ax              7) НЕТ, число бит - БЕЗЗНАКОВОЕ
    mov X, ebx
endif
       endm
```

ОЦЕНКА=2

---

4. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

```
Nomer record x:2,y:2,z:7
.data
X Nomer <>                       1) По условию A Nomer <>
.code
      mov ax,X
      and ax,mask y; mask y=00110000000b
      shr ax,y; —> <0,0,y>
      outword ax
```

ОЦЕНКА=5

---

5. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово **(dw)** с именем Param, описанной в головном модуле.

```
.data
                           1) public Param
Param dw ?
.code
      extrn F@0:near
Start:
      call F@0;
                           2) F - функция, куда её ответ ?
      exit
      end Start;
```

ОЦЕНКА=2

108-№6. Ибрагимова С.В.

1. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

2. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

3. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

4. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

3.

```
Nomer record X:2,Y:6,Z:7
A Nomer <>
; печать Y
      mov ax, A ;  потому что 15 битов и влезет
      and ax, mask Y
      shr ax,Y
      outint ax          2) outword
```

ОЦЕНКА=6

4.

```
; № = 6
.data
      extern Perem: word
.code
      public F
; prologue
Start:
F proc
      push eax
      push ebx
      mov eax, Param
      mov bx, 6 ; №
      div bx
      mov Perem, ax
      pop ebx
      pop eax
endp                          1) F endp
end Start                     2) Не понято условие задачи
```

" вызывает функцию без параметров F, описанную в каком-то другом модуле "

ОЦЕНКА=0

1.

```
include console.inc
N equ 2000
.data
X dw ?                        1) Не понято условие, X - массив !
Y dd ?
.code
```

```
Sum proc
      push ebp
      mov   ebp, esp
      push eax
      push ebx
      push ecx
      push edx
      push edi
      push esi
      mov  ebx, [ebp + 8]; адрес начала массива X
      mov  ecx, [ebp + 12]; N
      mov  edx, [ebp + 16]; сюда кладём Y
      xor  edi, edi ; счётчик
      mov  di, word ptr [ebx + 2*ecx - 4]; берём предпоследний элемент
                                  2) Не надо word ptr
      outstr       предпоследний элемент  это '
                                  3) ????
      outintln di
      xor esi, esi
L:
      mov ax, word ptr [ebx]      2) Не надо word ptr
      cmp ax, di
      jae N                       4) НЕТ, знаковые числа !
      add si, ax                  5) СУММА - dd !
N:
      add ebx, 2
      dec ecx
      cmp ecx,0                   6) Не надо, уже есть флаги
      ja L
finish:                           7) finish НИГДЕ не используется

      mov dword ptr [edx], esi; Y := Сумме
                                  8) НЕТ, у ВАС сумма в si !
      pop esi
      pop edi
      pop edx
      pop ecx
      pop ebx
      pop eax
      pop ebp
      ret 3*4
Sum endp
Start:
      push offset Y
      push N
      push offset X
      call Sum
      outstr ' Y равен '
      outint Y
      exit
end Start
```

ОЦЕНКА=1

2.

```
include console.inc
Probit macro X:req

                                  0) local !!!

for i <al,ah,bl,bh,cl,ch,dl,dh,ax,bx,cx,dx,si,di,bp,sp,eax,ebx \
       ecx,edx,esi,edi,esp,ebp>;
                                  1) Зачем определять регистр ????
                                  2) В цикле for i НЕ ИСПОЛЬЗУЕТСЯ
```

```
  if type X EQ 1
      mov al, X
      mov cl, 8
      xor ebx, ebx ; счётчик
L1: shr al, 1; остаток будет на CF
      adc bl, 0; суммируем остатки, то есть найдём количество единиц
      Loop L1
      mov dl, 8
      sub dl, bl; edx := 8 - ebx
      mov al, bl
      mul dl; ax := al*bl
      mov X, al
      endif
if type X EQ 2
      mov ax, X
      mov cl, 16
      xor ebx, ebx
L2: shr ax, 1
      adc bl, 0
      Loop L2
      mov dl, 16
      sub dl, bl
      mov al, bl
      mul dl
      mov X, ax
      endif
if type X   EQ    4
      mov eax, X
      mov cl, 32
      xor ebx, ebx
L3: shr eax, 1
      adc bl, 0
      Loop L3
      mov dl, 32
      sub dl, bl
      mov al, bl
      mul dl
      movzx X, ax
      endif
endm
.data
      Y db 11100000b
.code
Start:
      Probit Y
      outstr ' Y = '
      outwordln Y
      exit
end Start
```

1) Второй раз !?

```
ProBit macro x:req
      local Q, R, N
if type x eq 0 or type x eq 8
```

2) А если type x = 3 или type x < 0 ???

3) **.err** !!!

```
      outstrln "Неверный тип параметра "
```

4) ПЕЧАТИ НИКОГДА НЕ БУДЕТ

```
      exitm
endif
if type x eq 4
      ifidn <x>, <eax, ebx, ecx, edx, esi, \
            edi, esp, ebp, eip>
        ifidn <x>,<eax>
```

5) Зачем определять регистр ?

```
      mov ecx, 32                          6) Макро НЕ ДОЛЖНО быть большим !
      mov esi, 1
      xor edx,edx
      xor ebx,ebx
N: test x,esi
JZ Q
inc edx ;0
JMP R
Q: inc ebx ;1
R:
shr x, 1
LOOP N
mov x, edx
mul ebx
endif
ifidn <x>,<ebx, edi, esp, ebp, eip>
mov ecx, 32
xor edx,edx
xor eax,eax
N: test x,x
JZ Q
inc edx ;0
JMP R
Q: inc eax ;1
R:
shr x, 1
LOOP N
mul edx
mov x, eax
endif
ifidn <x>,<edx>
mov ecx, 32
mov esi, 1
xor eax,eax
xor ebx,ebx
N: test x,esi
JZ Q
inc ebx ;0
JMP R
Q: inc eax ;1
R:
shr x, 1
LOOP N
mul ebx
mov x, eax
endif
ifidn <x>,<esi>
mov ecx, 32
mov edi, 1
xor eax,eax
xor ebx,ebx
N: test x,edi
JZ Q
inc ebx ;0
JMP R
Q: inc eax ;1
R:
shr x, 1
LOOP N
mul ebx
mov x, eax
endif
ifidn <x>,<ecx>
mov ebx, 32
```

```
mov esi, 1
xor edx,edx
xor eax,eax
N: test x,esi
JZ Q
inc edx ;0
JMP R
Q: inc eax ;1
R:
shr x, 1
dec ebx
cmp ebx, 0
JA N
mul edx
mov x, eax
endif
else
mov ecx, 32
mov esi, 1
xor eax,eax
xor ebx,ebx
mov edx, x
N: test edx, esi
JZ Q
inc ebx ;0
JMP R
Q: inc eax ;1
R:
shr edx, 1
LOOP N
mul ebx
mov x, eax
endif
elseif type x eq 2
ifidn <x>, <ax, bx, cx, dx, si, di, sp, bp>
ifidn <x>,<ax>
mov ecx, 16
mov si, 1
xor dx,dx
xor bx,bx
N: test x,si
JZ Q
inc dx ;0
JMP R
Q: inc bx ;1
R:
shr x, 1
LOOP N
mov x, dx
mul bx
endif
ifidn <x>,<bx, di, sp, bp>
mov ecx, 16
mov si, 1
xor dx,dx
xor ax,ax
N: test x,si
JZ Q
inc dx ;0
JMP R
Q: inc ax ;1
R:
shr x, 1
LOOP N
```

```
mul dx
mov x, ax
endif
ifidn <x>,<si>
mov ecx, 16
mov di, 1
xor ax,ax
xor bx,bx
N: test x,di
JZ Q
inc bx ;0
JMP R
Q: inc ax ;1
R:
shr x, 1
LOOP N
mul bx
mov x, ax
endif
ifidn <x>,<dx>
mov ecx, 16
mov si, 1
xor ax,ax
xor bx,bx
N: test x,si
JZ Q
inc bx ;0
JMP R
Q: inc ax ;1
R:
shr x, 1
LOOP N
mul bx
mov x, ax
endif
ifidn <x>,<cx>
mov bx, 16
mov si, 1
xor dx,dx
xor ax,ax
N: test x,si
JZ Q
inc dx ;0
JMP R
Q: inc ax ;1
R:
shr x, 1
dec bx
cmp bx, 0
JA N
mul dx
mov x, ax
endif
else
mov cx, 16
mov si, 1
xor ax,ax
xor bx,bx
mov dx, x
N: test dx, si
JZ Q
inc bx ;0
JMP R
Q: inc ax ;1
```

```
R:
shr dx, 1
LOOP N
mul bx
mov x, ax
endif
elseif type x eq 1
ifidn <x>, <al, bl, cl, dl, ah, bh, ch, dh>
ifidn <x>,<al>
mov ecx, 8
mov dh, 1
xor dl,dl
xor bl,bl
N: test x,dh
JZ Q
inc dl ;0
JMP R
Q: inc bl ;1
R:
shr x, 1
LOOP N
mov x, dl
mul bl
endif
ifidn <x>,<bl, ah, bh>
mov ecx, 8
mov dh, 1
xor dl,dl
xor al,al
N: test x,dh
JZ Q
inc al ;0
JMP R
Q: inc dl ;1
R:
shr x, 1
LOOP N
mul dl
mov x, al
endif
ifidn <x>,<dh>
mov ecx, 8
mov bh, 1
xor al,al
xor bl,bl
N: test x,bh
JZ Q
inc al ;0
JMP R
Q: inc bl ;1
R:
shr x, 1
LOOP N
mul bl
mov x, al
endif
ifidn <x>,<dl>
mov ecx, 8
mov dh, 1
xor al,al
xor bl,bl
N: test x,dh
JZ Q
inc al ;0
```

```
JMP R
Q: inc bl ;1
R:
shr x, 1
LOOP N
mul bl
mov x, al
endif
ifidn <x>,<cl, ch>
mov ebx, 8
mov dh, 1
xor dl,dl
xor al,al
N: test x,dh
JZ Q
inc dl ;0
JMP R
Q: inc al ;1
R:
shr x, 1
dec ebx
cmp ebx, 0
JA N
mul dl
mov x, al
endif
else
mov ecx, 8
mov dh, 1
xor al,al
xor bl,bl
mov dl, x
N: test dl, dh
JZ Q
inc al ;0
JMP R
Q: inc bl ;1
R:
shr dl, 1
LOOP N
mul bl
mov x, al
endif
endif
endm
```

(это второй вариант 2 и он учитывают больше вариантов…)

_) УЧИТЬ МАКРОСЫ

ОЦЕНКА=0

108-№8. Какунин К.В.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.

```
001   ВВЦ 100 208 000
002   МОД 000 100 022; мод 000 000 002
                                                  1) СНАЧАЛО - проверка на НОЛЬ !
                                                     а не на ЧЁТНОСТЬ
003   ПНР 000 007 000; вовзрат
004   СЛЦ 023 023 100
005   СЛЦ 000 100 100                             2) <0>:=2*X[i] ???
                                                     Надо сложение с НУЛЁМ
                                                     2*X[i] > Maxlongint
006   ПР  000 013 000 проверка
007   СЛЦ 002 002 020 + 000 001 000
008   СЛЦ 004 004 019 + 000 000 001
009   СЛЦ 005 005 021 + 000 001 001
010   ВЧЦ 000 024 005  конец массива
011   ПР  000 013 000
012   БЕЗ 000 002 000
013   СЛЦ 000 023 023                             3) Надо сложение с НУЛЁМ
014   ПР  000 017 000
015   ВЫЦ 023 001 000
016   БЕЗ 000 018 000
017   ВЫЦ 019 001 000
018  СТОП 000 000 000
019       000 000 001                             4) Где КОП ??
020       000 001 000                             4) Где КОП ??
021       000 001 001                             4) Где КОП ??
022       000 000 002                             4) Где КОП ??
023       000 000 000; сум              4) Где КОП ??
024       000 207 207                             4) Где КОП ??
                                                  5) Надо СЛЦ 000 207 207
```

ОЦЕНКА=2

---

2. Выписать вид внутреннего машинного представления целой переменной X в 16-м виде

```
X dw -(15*G+2-№); G - номер Вашей группы
X dw -1626
```

```
0000 1110 0101 1010
1111 0001 1010 0101
+                 1
1111 0001 1010 0110   Answer: A6F1h
```

1) 0A6F9h

ОЦЕНКА=0

---

3. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G - номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

```
PROCN3 proc
  push EBP
  mov  EBP, ESP
  push EAX
  push EBX
  push ECX
  push EDX
  push ESI
  push EDI
  mov  EAX, [EBP + 8]; ^X
```

```
  mov EBX, [EBP + 12]; N
  lea EDX, [EAX][2 * EBX] - 4;
      mov DI, [EDX];
  mov ESI, 0                                    1) sub
     mov ECX, 0                                 1) sub
L:
  cmp word ptr [EAX][2 * ECX], DI          2) Не надо word ptr
  jge N
  movsx EDX, word ptr [EAX][2 * ECX]
  add ESI, EDX
N:
  inc ECX
  cmp ECX, EBX
  jnz L
  mov EAX, [EBP + 16]
  mov [EAX], ESI
  pop EDI
  pop ESI
  pop EDX
  pop ECX
  pop EBX
  pop EAX
     pop EBP
  ret 3 * 4
PROCN3 endp
Start:
  push offset Y
  push 10
  push offset MAS
  call PROCN3
  outintln Y
  exit
end Start
```

ОЦЕНКА=6-

---

4. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

ОЦЕНКА=0

---

5. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

```
mov EBX, dword ptr A
and EBX, mask Y
shr EBX, Y
outwordln EBX
```

ОЦЕНКА=6

108-№9. Клоков Т.А.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.
2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.
3. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G – номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

---

1)
Матрица в памяти представлена линейно → займем клетки 101-325; 9 div 2 = 4 столбец, a14 в 104 клетке, остальные отличаются на 15

```
1       ВВЦ  101 225 000; считывание массива
2       СЛЦ  011 011 104; S := S + x[i], где x[i] – элемент из 4 столбца
3       СЛЦ  002 002 010; i := i + 15
4       СЛЦ  012 012 009; K := K + 1
5       ВЧЦ  000 012 010;<0>:=K - 15
6       УСЛ  007 002 007; if K >= 15 then jmp 007, просмотрели все строки
7       ВЫЦ  011 001 000; write(S)
8       СТОП 000 000 000
9       00   000 000 001; const 1
10 00    000 000 015; const 15
11 00    000 000 000; S:= 0
12 00    000 000 000; K:=0
```

ОЦЕНКА=6

---

2)

```
include console.inc
.data
X dd ?
.code
start:
       inint X
       xor eax, eax; счетчик
       xor ecx, ecx; текущая позиция
       mov ebx, X
Loop_l:
       shr ebx, 1
       jnc L1
       add eax, ecx
L1:
       inc ecx
       cmp ecx, 32
       jb  Loop_l
       outword eax
                              1) exit
end start
```

ОЦЕНКА=5

---

3)

Спецификация: сумма чисел может вылезти за пределы 4 байт. В этом случае выдаем ошибку на экран и завершаем счет.

```
include console.inc
.data
N equ 1000108
Mas dw N dup (?)
Y dd ?
.code
```

```
F proc
       push ebp
       mov  ebp, esp
       push eax
       push ebx
       push ecx
       push edx
       push esi
       xor  esi, esi
       mov  ecx, [ebp + 12]; N
       sub  ecx, 2
       add  ecx, ecx
       mov  ebx, [ebp + 8]; ^X
       movzx edx, word ptr [ebx + ecx]; предпоследний

                                       1) movzx edx,word ptr [ebx+2*ecx-4]
;      add  ecx, 4
       Loop_l:
       movzx eax,word ptr [ebx+ecx]
       cmp  eax, edx
       jbe  L1
       add  esi, eax
       jc E_l
L1:
       sub ecx, 2
       cmp ecx, -2
       jg Loop_l                       2) НЕТ, ja
       mov  ebx, [ebp + 16]
       mov  [ebx], esi
       jmp  End_l
E_l:; слишком большое число
       outstrln "Too big value!"
End_l:
       pop esi
       pop edx
       pop ecx
       pop ebx
       pop eax
       pop ebp
       ret 12
F endp
start:
       push offset Y
       push N
       push offset Mas
       call F
                                       1) exit
       end start
```

ОЦЕНКА=4

108-№10. Марченко Ф.А.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.
2. Выписать вид внутреннего машинного представления целой переменной X в 16-м виде

```
X dw -(15*G+2-№); G – номер Вашей группы
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **dd**, и выводит ближайшее к X целое число, которое больше X и кратно №. Если нужное число больше `MaxLongInt`, то вывести "Большое число". Например, для `X=-18` и `№=12` надо вывести ответ `-12`.
4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 0, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.
5. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

108-№11. Муравицкая Е.Я.

1. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 1, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.
№ = 11 => chr(11)=11

```
Include console.inc
.code
Start:
      xor ecx, ecx ;sum
L:
      xor eax,eax
      Inchar al
      Cmp al,'.'
      jE  fin;    нет chr(11)
      cmp al, 11
      jE  next1
      cmp al,'0'
      jb  L
      cmp al,'9'
      ja  L
      sub eax,'0' ;ord( c ) - ord(0)
      test eax,1
      jNZ L
      add ecx, eax
      jmp L
fin:
      mov ecx, 1
next1:
      outword ecx
      exit
End Start
```

ОЦЕНКА=6

---

2. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G - номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

```
Sum proc
      Push ebp
      Mov  ebp, esp
      Push eax
      Push ecx
      Push edx
      Push esi
      Mov esi, [ebp+8] ; offset X
      Mov ecx, [ebp+12] ; N
      Xor eax,eax
      Mov dx, [esi+2*ecx-4]
L:
      Cmp dx, [esi]
      jBE next
      add eax, word ptr [esi]       1) НЕЛЬЗЯ dd+dw !
next:
      add esi, 2
      loop L
```

```
      mov ecx, [ebx+16] ; Y
      mov [ecx], eax
      pop esi
      pop edx
      pop ecx
      pop eax
      pop ebp
      ret 12
Sum endp
-----------------------
      Push offset Y ; адрес Y
      Push N
      Push offset X ; адрес X - массив типа MAS
      Call Sum
```

ОЦЕНКА=4

---

3. Дано описание

```
Nomer record A:3,B:№,C:4
X Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле B переменной X.

№ = 11 => Nomer record A:3, B:11, C:4
X Nomer <>

Фрагмент, который печатает поле B переменной X:

```
      Mov eax, X
      And eax, mask B
      Shr eax, B
      Outint eax        1) outword
```

ОЦЕНКА=5

108-№12. Муравский Д.П.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.
2. Выписать вид внутреннего машинного представления целой переменной X в 16-м виде

```
X dw -(15*G+2-№); G – номер Вашей группы
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое знаковое число X в формате **dd**, и выводит ближайшее к X целое число, которое больше X и кратно №. Если нужное число больше `MaxLongInt`, то вывести "Большое число". Например, для `X=-18` и `№=12` надо вывести ответ `-12`.
4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *нечётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 0, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.
5. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

108-№16. Сахаутдинова А.Р.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.
2. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.
3. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.
4. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

№3.

```
push eax
mov  eax, A
and  eax, mask Y
shr  eax, Y
outword eax
pop  eax
```

ОЦЕНКА=6

№4.

```
include console.inc
      public Param
      F proto
.data
Param dw 123
.code
Start:
      call F            1) Не понято, F - функция,
                           что сделано с её результатом ?
      pause
      exit
      end Start
```

ОЦЕНКА=3

№2.

```
include console.inc
.data
X db 12
.code
ProBit macro X:Req
      local n
            local z
            n = 0
            z = 0
if type X eq 0                      1) A если type X=3 или меньше нуля ?
            .err <Wrong operand type>
else
      repeat type X*8
            local L
            outwordln eax           2) ?????
            pol X, 1                3) pol - НЕТ ТАКОГО
```

```
            jc  L
            n = n + 1               4) ЭТО будет ВСЕГДА !
                                       ПОНЯТЬ ОБЯЗАТЕЛЬНО
            z = z - 1               5) ???
L:
      z = z + 1
endm
      mov X, n*z
endif
endm                            6) На этапе компиляции X
                                       НЕ СУЩЕСТВУЕТ

Start:
mov eax, 12345
ProBit eax
outwordln eax
pause
exit
end Start
```

ОЦЕНКА=0

№1.

```
001—ВВЦ—100—315—000 ; ВВОД МАССИВА        1) Не понято условие
                                                 "X длины 200+№"
002—ВЧЦ—000—100—016
003—УСЛ—012—004—004                    2) Не понято условие
                                              Когда 0 ЕСТЬ,
                                              Печать суммы, ИНАЧЕ
                                              Печать 1
004—МОД—000—100—018; S:= x[i] mod 2
005—УСЛ—006—008—008
006—ДЕЦ—014—014—018                    3) Зачем СУММУ делить на 2 ?
007—СЛЦ—004—004—100                    4) ??? Испорчена команда 004
008—СЛЦ—004—004—015                    5) Нет цикла на 216 раз
009—СЛЦ—002—002—015
010—СЛЦ—007—007—017
011—БЕЗ—000—002—000
012—ВЫЦ—014—001—000
013—СТОП-000—000—000
014—00— 000—000—001; сумма
015—00— 000—001—000; i := i +1
016—00— 000—000—000; const = 0
017—00— 000—000—001; const = 1
018—00— 000—000—002; const = 2
```

ОЦЕНКА=0

108-№15. Савицкий И.П.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.

3. Дано описание

```
Nomer record A:3,B:№,C:4
X Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле B переменной X.

4. Написать на Ассемблере головной модуль, который вызывает процедуру без параметров P, описанную в каком-то другом модуле. Эта процедура должна делить на № значение беззнаковой переменную размером в слово (**dw**) с именем Perem, описанную в головном модуле.

ЗАДАНИЕ 1

001	ВВЦ	100	225	000	Ввод 15x15 элементов матрицы (лежит она на 100 - 324)
002	СЛЦ	012	012	101	Прибавляем к S очередной элемент матрицы (101 — первый такой для 15 mod 2 = 1 столбца) **1) Не понято условие, 15 div 2 = 7**
003	СЛЦ	002	002	010	Прибавляем к сложению 15 чтобы перейти на следующую строку
004	СЛЦ	011	011	009	i := i +1
005	ВЧЦ	000	011	010	i < 15? (потому что столбцы нумеруется 0..14)
006	ПМ	000	002	000	Если да - прыгаем по циклу
007	ВЫЦ	012	001	000	Вывод S
008	СТОП	000	000	000	Остановка
009	0	000	000	001	Константа 1
010	0	000	000	015	Константа 15
011	0	000	000	000	i
012	0	000	000	000	S

ОЦЕНКА=2

ЗАДАНИЕ 2

```
.data
       X dd ?
.code
Start:
       inint X
       mov eax, 1
       mov bl, 0
       mov ecx, 31
       xor edx, edx
L:
       test X, eax
       jz  continue
       add edx, bl                1) НЕЛЬЗЯ dd+db !
continue:
       inc bl
       shl eax                    2) shl eax,1
       loop L
       outword edx
       exit
end Start
```

ОЦЕНКА=2

ЗАДАНИЕ 3

```
.code
Start:
...(какой-то код со вводом)
       mov eax, X
       and eax, mask B
       shr eax, B
       outword eax
       …
end Start
```

ОЦЕНКА=6

ЗАДАНИЕ 4

Головной модуль:

```
.data
       public Perem
       Perem dw ?
.code
       extern P
Start:
       call P
       exit
end Start
```

Вспомогательный модуль:

```
.data
       extern Perem: word
.code
public P
P proc
       push eax
       push ebx
       mov  eax, Perem           1) НЕЛЬЗЯ, dd <> dw
       mov  bx, 15
       div  bx                   2) УЧИТЬ ДЕЛЕНИЕ, dx=?
       mov Perem, ax
       pop ebx
       pop eax
       ret
P endp
```

ОЦЕНКА=2

108-№21. Шамков И.В.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X

```
X dw -(15*G+1-№); G – номер Вашей группы
```

3. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.

4. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 1, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.

5. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G – номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

№2

```
-(15*108 + 1 – 21) = -1600
     00000110 01000000b
     11111001 10111111b; обратный код
   +00000000 00000001b
     1111 1001 1100 0000b
     0F9C0h
Ответ: 0C0F9h
```

ОЦЕНКА=6

№5

```
include console.inc
; выводит сумму тех цифр которые больше предпоследнего
; word значит числа беззнаковые
N equ 10008
.data
X dw N dup (?)
Y dd ?
.code
Sum proc
      push ebp
      mov ebp, esp
      push eax
      push ebx
      push ecx
      push edx
      push edi
      push esi
      mov ebx, [ebp + 8]; адрес начала массива X
      mov ecx, [ebp + 12]; N
      mov edx, [ebp + 16]; сюда кладём адрес Y
      xor edi, edi
      mov di,word ptr[ebx+2*ecx-4]; di:=X[n-1]       1) Не надо word ptr
      xor esi, esi
L:    mov ax, word ptr [ebx]                         1) Не надо word ptr
      cmp ax, di; сравниваем тек эл-т с предпоследним
      jbe not_sum
      movzx eax, ax
      add esi, eax;
```

```
not_sum:
      add ebx, 2
      dec ecx
      cmp ecx,0                                          2) Не надо
      ja L
finish:
      mov dword ptr [edx], esi; Y := Summa               3) Не надо dword ptr
      pop esi
      pop edi
      pop edx
      pop ecx
      pop ebx
      pop eax
      pop ebp
      ret 3*4
Sum endp
Start:
      push offset Y
      push N
      push offset X
      call Sum
      outstr ' Y = '
      outint Y                                           4) outword
      exit
end Start
```

ОЦЕНКА=5

№1

```
Сумма 10 столбца
1     ВВЦ  100 225 000 ; ввели элементы матрицы в 100-324
2     СЛЦ  011 011 109; 109  - первый элемент 10 столбца
2     СЛЦ  002 002 009; сдвигаем на 15 указатель на эл-т 10 столбца
4     СЛЦ  010 010 012; считаем сколько раз сдвинулись
5     ВЧЦ  000 010 013;  пока не прибавили 15 раз идем дальше
6     ПМ   000 002 000
7     ВЫЦ  011 001 000; writeln(s)
8     СТОП 000 000 000; окончание работы
9     00   000 000 015;
10    00   000 000 000; i := 0 изначально счетчик до 15
11    00   000 000 000; тут сумма s :=0 изначально
12    00   000 000 001; константа 1
13    00   000 000 016;
```

ОЦЕНКА=6

№4

```
include console.inc
.code
Start:
      xor edx, edx; в edx счётчик №
      xor esi, esi; в esi сумма
      xor ebx, ebx; вспомогательная
Rep_inp:
      inint al
      cmp al, '.'
      je finish
      cmp al, '№'                1) Не понято условие №=21
      jne not_inc
      inc edx
```

```
not_inc:
      cmp al, '0'
      jb  not_sum
      cmp al, '9'
      ja  not_sum; если не цифра то уже не суммируем
      sub al, '0'; в al осталось значение цифры то есть ord(digit) - ord(0)
      push eax; спасаем нашу цифру
      shr al, 1; проверяем на чётность
      jc not_sum
      pop eax; вытаскиваем из стэка спасённую цифру
      cmp edx, 0
      ja not_sum; тоесть если ещё нету №
      movzx eax, al; беззнаково расширили al до eax
      add esi, eax
      mov edi, 1; индикатор что такая цифра есть
      not_sum:
      jmp Rep_inp
finish:
      cmp edi, 0
      jne print_Sum
      mov esi, 1
print_Sum:
      outword esi
      exit
      end Start
```

ОЦЕНКА=4

№3

```
include console.inc
.data
; N = 21
; 21 mod 3 = 0
      X dd ?
.code
Start:
      inint X
      mov eax, X
      xor esi, esi; здесь сумма
      mov ecx, 32
L:    shr eax, 1; тоесть (?) смотрим правый бит
      jnc not_sum; тоесть в CF 0
      add esi, ecx; правда запишется на 1 больше нужного
```

1) Не понято условие: ПРАВЫЙ бит
   имеет номер ноль !
   учить АРХИТЕКТУРУ

```
      dec esi;
not_sum:
      Loop L
      outword esi
      exit
end Start
```

ОЦЕНКА=2

108-№18. Скворцова А.Ю.

1. Пусть на Паскале дано описание типа массива:

```
const N=2000 G; {G – номер Вашей группы};
type MAS=array[1..N] of integer;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**. Процедура должна присвоить параметру Y сумму элементов этого массива, которые меньше его предпоследнего элемента. Привести пример вызова этой процедуры.

2. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов r8, m8, r16, m16, r32 или m32, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

3. Дано описание

```
Nomer record X:2,Y:№(18),Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

4. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

№1:

```
push offset Y
push N
push offset X
call P

P proc
   push ebp
   mov  ebp, esp
   push eax
   push ebx
   push ecx
   push edx
   push esi
   mov  eax, [ebp+8] ; offset x
   mov  ecx, [ebp+12] ; n
   mov  ebx, [ebp+16] ; offset y
   xor  edx, edx
   mov  [ebx], edx ; y := 0
   mov  esi, eax
   add  esi, ecx
   add  esi, ecx   ; складываем два раза, т.к. один элемент занимает 2 байта
   sub  esi, 2; получили ссылку на предпоследний элемент
                              1) sub  esi, 4; !
                              2) ПРОСТО lea esi,[eax+2*ecx-4]
L:
   mov  dx, [eax]
   cmp  dx, [esi]
   jae  T                     3) НЕТ, знаковые !
   add  [ebx], edx
T:
   add eax, 2
   loop L
   pop esi
   pop edx
   pop ecx
   pop ebx
   pop eax
   pop ebp
   ret 3*4
P endp
```

ОЦЕНКА=4

№3:

```
mov     eax, A
and     eax, mask Y;
shr     eax, Y;      eax:=<0,0,Y>
outword eax
```

ОЦЕНКА=6

№4:

Головной модуль:

```
include console.inc
.data
   public Param
   Param dw (?)
.code
   extern F                                  1) F@0
begin:
   call F
                                             2) Куда ответ ФУНКЦИИ ???
end begin
```

Вспомогательный модуль: (на всякий случай)

```
include console.inc
.data
   extern Param: word
.code
   public F
F proc
   push ebp
   mov ebp, esp
   pushad
   mov cx, 18
   mov dx, 0                                 3) ЭТО НЕ ДЕЛЕНИЕ !!!
                                             4) НЕТ, Знаковые
   mov ax, Param
   imul cx
   mov Param, ax                             5) Не понято условие задачи
                                                "вызывает функцию без параметров F"
   popad
   pop ebp
F endp
   End
```

ОЦЕНКА=2

№2:

```
ProBit macro x: req
   local L, K, T    ; совершенствуюсь в своем познании и использую один набор меток
   xor eax, eax
   xor ebx, ebx
if type x EQ 1
    mov cl, x
L:
    shr cl, 1
    jnc K
    add al, 1
    jmp T
K:
    add bl, 1
T:
    cmp cl, 0
    ja L
    mul bl
    mov x, al
elseif type x EQ 2
    mov cx, x
L:
```

```
    shr cx, 1
    jnc K
    add al, 1
    jmp T
K:
    add bl, 1
T:
    cmp cx, 0
    ja L
    mul bl
    mov x, ax
elseif type x EQ 4
    mov ecx, x
L:
    shr ecx, 1
    jnc K
    add al, 1
    jmp T
K:
    add bl, 1
T:
    cmp ecx, 0
    ja L
    mul bl
    mov x, eax
else
   .err <Bad argument!>
   exitm
endif
   endm
```

ОЦЕНКА=6

108-№19. Толеутаева А.Б.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа chr(№), или 1, если такого символа в тексте нет. Считать, что это произведение не более MaxLongword.

4. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G – номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

5. Написать макроопределение с заголовком

```
RazBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов r8, r16, m8, m16, r32 или m32, печатать (по **outint**) разность количества бит со значением "1" и количеством бит со значением "0" во внутреннем машинном представлении параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики об ошибках обращения и неверных типах своего операнда.

№1

№ div 2 = 19/2 = 8; 15*15 = 225

```
1  ВВЦ 100 225 000 ; Read mas
2  СЛЦ 016 107 016 ; S:=S+x[i,j]. В начале элемент в ячейке 107, затем 107 + 15...
3  ВЧЦ 017 017 015 ; i - 1
4  ПР  000 011 000 ; i = 0 -> вывод и выход
5  СЛЦ 002 014 002 ; Продолжаем цикл, переход на следующую строчку
6  БЕЗ 000 002 000 ;
7  000 000 000 000                 1) ??????
8  000 000 000 000                 1) ??????
9  000 000 000 000                 1) ??????
10 БЕЗ 000 012 000                 2) Сюда НИКОГДА не придём !!!
11 ВЫЩ 016 001 000
12 СТОП 000 000 000
13 00 000 000 000;
14 00 000 015 000; const 15 для индекса массива
15 00 000 000 001; const 1
16 00 000 000 000; Sum
17 00 000 000 015; i
```

ОЦЕНКА=4

№2

2. Написать полную программу на Ассемблере, которая вводит (по **inint**) целое число в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Программа выводит (по **outword**) *сумму номеров* тех позиций во внутреннем представлении числа, на которых стоят биты со значением "1" (самый правый бит числа имеет номер 0). Например, для 0100110b надо вывести ответ 1+2+5=8.

N mod 3 = 19 mod 3 = 1 => dw

```
.686
include console.inc
.data
T dw ? ;16
.code
Start:
   Inint T ;целое знаковое       1) Для этой задачи БЕЗРАЗЛИЧНО !
   shl T, 1                      2) Потерян бит с номером 31 !
   Sub ecx, ecx
   Sub eax, eax; answer
cycle:
```

```
      Shl T, 1
      Jnc next; not 1
      Add eax, ecx                    3) Не понято условие
                                         " самый правый бит числа имеет номер 0"
   next:
      inc ecx
      cmp ecx, 15
      jb  cycle
       outword eax
       exit
       end Start
```

ОЦЕНКА=2

№3

3. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 1, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.

```
СЧИТАЕМ chr(19) СЛУЖЕБНЫМ, не прибавляем к сумме!!!
                                  1) По условию ДО этого символа, именно
                                     поэтому не прибавляет
include console.inc
.code
Start:
      sub ebx, ebx ;sum
cycle:
      inchar cl
      cmp cl, '.'
      je prov
      cmp cl, 19
      je next
      cmp cl, '0'
      jb  cycle ;не цифра
      cmp cl, '9'
      ja  cycle ;не цифра
      movzx ecx,cl
      sub ecx, '0'
      test ecx, 1
      jnz cycle ;нечетное
      add ebx, ecx
      jmp cycle
prov:
      mov ebx,1
next:
      outword ebx
      exit
      end Start
```

ОЦЕНКА=6

№4

4. Пусть на Паскале дано описание типа массива:

```
const N=180000;                       1) N=108000
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

```
   P proc
      Push ebp
      Mov  ebp, esp
      Pushad
```

```
   Mov eax, [ebp + 8]; ^X
   Sub ebx, ebx ;обнуляем старшие байты
   Mov bx, word ptr [ebp+12] ;N вмещается в word
                                1) Не надо word ptr
                                2) Процедура НЕ ЗНАЕТ, что
                                   "N вмещается в word"
   Mov edx, offset [eax + 2*ebx - 4] ;адрес предпоследнего
                                3) ОШИБКА, только "offset <имя>"
                                4) Mov dx,[eax+2*ebx-4]; САМ X[N-1]
   Mov di, word ptr [edx]; предпоследний
                                5) НЕТ
   Sub esi, esi ;summ
   Sub ecx, ecx
L: cmp word ptr [eax+2*ecx],di  6) Не надо word ptr
   Jbe R
   Sub edx, edx ;обнуляем старшие регистры, чтобы не портить сумму
   Mov dx,word ptr [eax+2*ecx]   6) Не надо word ptr
   Add esi, edx
R:
   Inc ecx
   Cmp ecx, ebx                 7) В ebx НИЧЕГО ХОРОШЕГО НЕТ
   Je L ;в начале ecx = 0, поэтому выходим при ecx = ebx
   Mov eax, [ebp + 16] ; offset Y
   Mov [eax], esi
    Popad
    Pop ebp
    Ret 12
P endp
```

Вызов:

```
   Push offset Y
   Push N
   Push offset MAS
   Call P
   Outint Y                     8) outword
```

ОЦЕНКА=2

№5

5. Написать макроопределение с заголовком

```
RazBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `r16`, `m8`, `m16`, `r32` или `m32`, печатать (по **`outint`**) разность количества бит со значением "1" и количеством бит со значением "0" во внутреннем машинном представлении параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики об ошибках обращения и неверных типах своего операнда.

```
include console.inc
.code
Razbit macro x
    local L, End_m, L2
    push ebx
    push ecx
                                1) А почему тогда нет push eax ?
    xor ebx, ebx
    xor ecx, ecx
if type x EQ 4
  ifdifi <x>, <eax>
    mov eax, x
  endif
L:
    cmp eax, 0
    je  End_m
    shr eax, 1
    adc ebx, 0; Число "1"
```

```
    inc ecx;
                                        2) Число ЗНАЧАЩИХ бит, не ДЛИНА X !
    jmp L
elseif type x EQ 2
  ifdifi <x>, <ax>
    mov ax, x
  endif
L:
    cmp ax, 0
    je End_m
    shr ax, 1
    adc ebx, 0
    inc ecx
    jmp L
elseif type x EQ 1
  ifdifi <x>, <al>
     mov al, x
  endif
L:
     cmp al, 0
     je End_m
     shr al, 1
     adc ebx, 0
     inc ecx
     jmp L
else
     echo 'Bad params'
     .err
     exitm
endif
End_m:
     sub ecx, ebx                   3) НЕ разность "0" и "1"
     cmp ecx, ebx
     ja L2
     xchg ebx, ecx                  3) Не понято условие
                "разность количества бит со значением "1" и количеством бит со значением "0""
                                      Она может быть < 0
L2:
    sub ecx, ebx
    outint ecx
    pop ecx
    pop ebx
    endm
start:
    mov al, 10
    Razbit al
    exit
end start
```

ОЦЕНКА=2

108-№20. Тунис И.С.

1. Написать программу для учебной машины УМ-3. Эта программа должна сначала вводить целочисленный массив X длины 200+№, затем печатать сумму чётных чисел этого массива, расположенных в массиве строго до первого нулевого элемента массива X. Если таких чисел нет, то вывести ответ 1. При записи кодов операций использовать мнемонические обозначения.

2. Выписать вид внутреннего машинного представления целой переменной X в 16-м виде

```
X dw -(15*G+2-№); G – номер Вашей группы
```

3. Написать макроопределение с заголовком

```
ProBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `m8`, `r16`, `m16`, `r32` или `m32`, присваивать параметру X новое значение, равное произведению числа бит со значением "1" и числа бит со значением "0" во внутреннем машинном представлении исходного параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики о неверных типах своего операнда.

4. Дано описание

```
Nomer record X:2,Y:№,Z:7
A Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле Y переменной A.

5. Написать на Ассемблере головной модуль, который вызывает функцию без параметров F, описанную в каком-то другом модуле. Эта функция возвращает увеличенное в № раз значение знаковой переменной размером в слово (**dw**) с именем Param, описанной в головном модуле.

№1

```
001 |ВВЦ   100 220 000|
002 |ВЧЦ   000 100 018|
003 |ПР     000 012 000|
004 |МОД   000 100 019|
005 |ПНР   000 007 000|
006 |СЛЦ   017 100 017|
007 |СЛЦ   002 002 021|
008 |СЛЦ   004 004 021|
009 |СЛЦ   006 006 021|
010 |ВЧЦ   000 002 020|
011 |ПНР   000 002 000|
012 |ВЧЦ   000 017 018|
013 |ПНР   000 015 000|
014 |СЛЦ   017 017 022|
015 |ВЫЦ   017 001 000|
016 |СТОП 000 000 000|
017 |00      000 000 000|     S:=0
018 |00      000 000 000|     const 0
019 |00      000 000 002|     const 2
020 |ВЧЦ   000 320 018|
021 |00     000 001 000|
022 |00     000 000 001|     const 1
```

ОЦЕНКА=6

№2

-(15*108+2-20)=-1602

0 1 1 0 0 1 0 0 0 0 1 0

1 0 0 1 1 0 1 1 1 1 0 1

\+ 1

1 0 0 1 1 0 1 1 1 1 1 0

9BE 1) НЕТ `0BEF9h`

ОЦЕНКА=0

№3

```
include console.inc
ProBit macro X
IFB <X>                                  1) Не понято условие X:Req
      .err <Wrong x>
ENDIF
IF (type X EQ 8) or (type X EQ 0)
      .err <Wrong x>                      2) А если type X=3 или <0 ???
ENDIF
      push EAX
      mov EAX, 0
REPEAT (type X) * 8
      local L1, L2
      shr  X, 1
      jc   L1
      inc  AL
      jmp  L2
L1:   inc  AH
L2:
ENDM
      mul  AH
      push EAX
      mov EAX, [ESP + 4]
      mov X, [ESP]
      add ESP, 8
ENDM

Start:
      mov BX, 3
.listmacro
      ProBit BX
.nolist
      outint BX
      pause
      exit
end Start
```

ОЦЕНКА=4

№4

```
Include console.inc
.data
       Nomer record x:2,Y:20,Z:7
       A Nomer <0,10,0>
.code
Start:
       mov EAX, A
       shr EAX, 10                       1) НЕТ, 7
                                         2) and        !!!
       outint EAX                        2) outword eax
       pause
end Start
```

ОЦЕНКА=0

№5

```
include console.inc
.data
Param dw ?
      extern F@0: proc
      public Param
.code
Start:
      call F@0
                                1) Не понято условие: F - функция,
```

```
                                    куда делся её ответ ?
    end Start
```

ОЦЕНКА=4

108-№5. Жуков Н.А.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 15x15 целых чисел и затем выводит сумму всех чисел, расположенных в столбце матрицы с номером № **div** 2. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая вводит текст до точки и выводит (по **outword**) сумму *чётных цифр* в этом тексте, которые расположены до первого символа `chr(№)`, или 1, если такого символа в тексте нет. Считать, что это произведение не более `MaxLongword`.

3. Пусть на Паскале дано описание типа массива:

```
const N=1000 G; {G – номер Вашей группы}
type MAS=array[1..N] of word;
```

Написать на Ассемблере процедуру со стандартными соглашениями о связях, которая получает в качестве параметров адрес массива типа MAS, его длину N и адрес целой переменной Y формата **dd**, которой процедура должна присвоить сумму элементов этого массива, больших его предпоследнего элемента. Привести пример вызова этой процедуры.

3. Написать макроопределение с заголовком

```
RazBit macro X:Req
```

Оно должно для своего параметра X, который может быть только форматов `r8`, `r16`, `m8`, `m16`, `r32` или `m32`, печатать (по **outint**) разность количества бит со значением "1" и количеством бит со значением "0" во внутреннем машинном представлении параметра X. Макроопределение должно настраиваться на тип параметра и выдавать необходимые диагностики об ошибках обращения и неверных типах своего операнда.

4. Дано описание

```
Nomer record A:3,B:№,C:4
X Nomer <>
```

Написать фрагмент на Ассемблере, который печатает поле B переменной X.

5. Написать на Ассемблере головной модуль, который вызывает процедуру без параметров P, описанную в каком-то другом модуле. Эта процедура должна делить на № значение беззнаковой переменную размером в слово (**dw**) с именем Perem, описанную в головном модуле.

N1(5 div 2=2)

```
001               ВВЦ                                          100 225 000
002               СЛЦ                                          009 101 009
003               СЛЦ                                          002 002 010
004               ВЧЦ                                          011 011 008
005               ПНР                                          000 002 000
006               ВЫЦ                                          009 001 000
007                    СТОП                                    000 000 000
008        00                                 000 000 001;        const        1
009            00                                    000 000 000;            Sum
010             00                                     000 015 000;            i
011 00    000 000 015;n
```

ОЦЕНКА=6

N2

```
include console.inc
.data
A DB ?
.code
Start:
   sub   esp,4
; flag - [esp]
; bx - answer                    1) НЕТ, ответ = dd
      mov byte ptr [esp],0
      mov bx,0                   2) ebx
@Lp:
      inchar dl
      mov byte ptr [A], dl       3) Это ПРОСТО mov A,dl  !!!
      cmp byte ptr [A],'.'       4) УЧИТЬ cmp A,'.'
                                    и чем dl ХУЖЕ A ??
      je  @endloop
      mov  al,byte ptr [esp]
      test al,al
      jne  @tsrp                        4) Был chr(5) ???
      cmp  byte ptr [A],'9'
      ja   @tsrp
      cmp  byte ptr [A],'0'
```

```
         jb   @tsrp
         movzx eax,byte ptr [A]          5)  Не  надо  byte  ptr,  ПОНЯТЬ  или
СПРОСИТЬ
         sub eax,'0'
         cdq                             6)  УЧИТЬ  отределять  чётность  БЕЗ
деления
         mov ecx,2
         idiv ecx
         cmp edx,0
         jne @tsrp
         movzx eax,byte ptr [A]
   ;      lea edx,dword ptr [eax-'0']    7) dword ptr БЕСПОЛЕЗНО
   ;      movsx eax,bx
   ;      add    eax,edx
   ;      mov bx,ax                      8) ПРОСТО add ebx,eax; ПОНЯТЬ
   @tsrp:
         cmp byte ptr [A],5
         jne @Lp
         mov byte ptr [esp],1
         jmp @Lp
   @endloop:
         mov  al,byte ptr [esp]
         test al,al
         je  @zer
         outword bx
         jmp @ex
   @zer:
         outword 1;
   @ex:
         add esp,4
         exit
   end Start
```

ОЦЕНКА=0

N3(массив)

```
include console.inc
N equ 108000
.data
X dw N                                   1) НЕЛЬЗЯ !!! N НЕ константа
                                         2) УЧИТЬ описывать МАССИВ
Y dd ?
.code
Pred proc
; type word =2
;[ebp + 8]; N                            3) НЕТ
;[ebp + 12]; Y                           3) НЕТ
;[ebp + 16]; mas                               3) НЕТ

   push ebp
   mov ebp, esp
   push eax
   push ebx
   push ecx
   push edx
   push edi
   push esi
   mov edx,[ebp + 8];  N
   mov ecx,[ebp + 12]; ^Y
   mov ebx,[ebp + 16]; ^X
   xor edi,edi
   xor esi,esi
```

```
   mov si,word ptr [ebx+2*edx-2*2]
@L:      mov ax, word ptr [ebx]              4) Не надо word ptr
   cmp ax, si
   jbe @next
   add di, ax                            5) НЕТ, сумма = dd
@next:
   add ebx, 2
   dec edx
   cmp edx,0                             6) НЕ НАДО
   ja  @L
   mov dword ptr [ecx], edi              7) В edi НЕТ суммы
   pop esi
   pop edi
   pop edx
   pop ecx
   pop ebx
   pop eax
   pop ebp
   ret 3*4
Pred endp

Start:
   push offset X                         3) НЕТ
   push offset Y                         3) НЕТ
   push N                                3) НЕТ
   call Pred
   exit
end Start
```

ОЦЕНКА=0

N4

```
Для Nomer record A:3,B:5,C:4
X Nomer <>
   xor eax,eax             1) Зачем ???
   mov ax,X
   and Ax,mask B
   shr Ax,B
   outu Eax                2) outword ax
```

ОЦЕНКА=5

N5

Головной

```
include settings.inc       1) include console.inc
                              Со   сдачей   программ   совсем   плохо   ?
   extern P@0:Proc
   public                                                          Perem
.data
   Perem                              dw                            ?
.code
   Start:
      call P@0
      exit
   end Start
```

Другой

```
include settings.inc          1) include console.inc
   extern Perem: word
```

```
   public                                                          P
.code
   P proc
     push eax
     push edx
     push ecx
     mov cx,5
     xor dx,dx
     mov ax,Perem
     div cx
     mov Perem,ax
     pop ecx
     pop edx
     pop eax
     ret
   P endp
     end
   ОЦЕНКА=4
```

107-№16. Петров П.В.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 10x20 *вещественных* чисел и выводит сумму номеров тех строк этой матрицы, которые начинаются с 0.0. При записи кодов операций использовать мнемонические обозначения.

2. Написать полную программу на Ассемблере, которая последовательно вводит (по **ininт**) целые знаковые числа, пока не будет введёно число № (признак конца ввода). Программа выводит сумму тех из этих чисел, которые чётные и без остатка делятся на №. Если сумма превышает `MaxLongInt`, то вывести "Большая сумма".

3. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (`T=`**dd** для № **mod** `3=0; T=`**dw** для № **mod** `3=1; T=`**db** для № **mod** `3=2)`. Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

4. Написать макроопределение с заголовком

```
NumOddNotWord macro X:VarArg
```

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов `m8`, `m16`, `m132` или `m64`. Макрос печатает число нечётных переменных из списка X, не являющихся словами (`m16`), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

5. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем `Perem_jmp` размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

107-№20. Сычев П.С.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 10x20 *вещественных* чисел и выводит сумму номеров тех строк этой матрицы, которые начинаются с 0.0. При записи кодов операций использовать мнемонические обозначения.

```
1   ввв     100     200     000
2   вчв     000     100     012
3   усл     004     005     005
4   слц     013     013     016     s=s+i
5   вчц     000     016     015     <0>=i-10
6   усл     010     007     007
7   слц     016     016     014     i=i+1
8   слц     002     002     017     j=j+1(delta)
9   без     000     002     000
10  выц     013     001     000
11  стоп    000     000     000
12  00      000     000     000     0
13  00      000     000     000     s
14  00      000     000     001     1
15  00      000     000     010     10
16  00      000     000     001     i
17  00      000     020     000     дельта
```

ОЦЕНКА=6

---

2. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для №**mod** 3=0; T=**dw** для №**mod** 3=1; T=**db** для №**mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

```
include console.inc
.data
  X db 14
.code
F proc
  push ebp
  mov esp, ebp
  push ebx
  push ecx
  push edx
  mov dl, 0
  mov ecx, 8
;  mov bx, word ptr [ebp+8]      1) НЕ НАДО word ptr
                                 2) mov bl,[ebp+8]
  mov dl, bl                     3) ЗАЧЕМ ???
                                    Ниже сразу sub dl,dl !
  dec ecx
L:
  sub dl, dl                     3) ЗАЧЕМ ???
                                    Ниже сразу mov dl,bl !

  mov dl, bl
;  shl dl, 5
```

```
;  shr dl, 5                    4) and dl,7 ???
  cmp dl, 7
  je  N                        5) Не понято условие задачи
                    "имеет младший бит "0" и (!) в X есть хотя бы три идущих подряд бита
  shr bl, 1
  loop L
  jmp net                      6) Нет проверки МЛАДШЕГО бита X !
N:
  mov eax, 0
  jmp konec
net:
  mov eax, 1
konec:
  pop edx
  pop ecx
  pop ebx
  pop ebp
F endp
start:
  sub  eax, eax
  mov  ebx, dword ptr X
  push ebx
  call F
  outint eax
end
```

ОЦЕНКА=2

---

3. Написать макроопределение с заголовком

`NumOddNotWord` **`macro`** `X:VarArg`

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов `m8`, `m16`, `m132` или `m64`. Макрос печатает число нечётных переменных из списка X, не являющихся словами (`m16`), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

```
NumOddNotWord macro X: VarArg
IFB <X>
      outint -1
ELSE
      push eax
      mov eax, 0
for a, <X>
      local no
  IF type(a) NE 2
      test byte ptr a, 1b
      jNz no
      inc eax
no:
  ENDIF
endm
      outint eax
      pop eax
ENDIF
      endm
```

ОЦЕНКА=6

---

4. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

```
include console.inc
        extern Perem_jmp: dword
.data
.code
Start:
        mov eax, Perem_jmp
        mov ebx, 3
        div ebx                     1) УЧИТЬ ДЕЛЕНИЕ! edx=?
L:      cmp edx, 0
        jNz L
        jmp [Perem_jmp]             2) jmp Perem_jmp
        pause                       3) ???
        exit                        3) ???
        end Start
```

ОЦЕНКА=3

---

5. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

CF:=OF **or** SF

```
include console.inc             1) Не понято условие "фрагмент"
.code
start:
        jo OF_1
;CF:=0
        mov al, 255
        cmp al, 250
        jmp end_start
OF_1:
        js OS_1
;CF:=0                           2) ;SF:=0
        mov al, 255
        cmp al, 250
        jmp end_start
OS_1:
;CF:=1
        mov al, 250
        cmp al, 255
end_start:
        exit
        end start
```

ОЦЕНКА=5

108-№4. Денисов В.М.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 10x20 *вещественных* чисел и выводит сумму номеров тех строк этой матрицы, которые начинаются с 0.0. При записи кодов операций использовать мнемонические обозначения.

```
01 ВВВ  101 200 000; Ввод матрицы
02 ВЧВ  000 101 014; <0>:=x[i]-0.0
03 УСЛ  004 005 005;if = →004, else →005
04 СЛЦ  016 016 015; S:= S+k
05 СЛЦ  002 002 013;I:= I+20;на след строку
06 СЛЦ  015 015 011;k := k-1                    1) k := k+1  ???
07 ВЧЦ  000 015 012; <0> := k-11
08 УСЛ  008 002 008; if k>=11 vihodim
09 ВЫЦ  016 001 000;Write(S)
10 СТОП 000 000 000
11 00   000 000 001; const 1
12 00   000 000 011; const 11
13 00   000 020 000; const 20
14<0,0>; вещ const 0                            2) 00 000 000 000 !!!
15 00   000 000 001;k
16 00   000 000 000;S
```

ОЦЕНКА=6

---

2. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

```
Include console.inc
.data
X dd 14; 2 + 4 + 8                      1) НЕТ X dw 14; !!!
.code
ArrSum proc
       push ebp
       mov ebp, esp
       push ebx
       push ecx
       push edx
       mov  bx, word ptr [ebp+8]        2) Не надо word ptr
                                           [ebp+8] = offset X !!!
       xor edx, edx
       mov dx, 1
       and dx, bx                       3) ПРОСТО test bx,1 !!!
       cmp dx, 0                        4) НЕ НАДО !!!
       je  No
       mov cx, 1
       mov dx, 0
       jmp L
Ad:
       inc dx
       cmp dx, 3
       je Yes
       shl cx, 1
L:
       cmp cx, 0
       je  No
       mov ax, bx
       and ax, cx
       cmp ax, 0
       jne Ad
       mov dx, 0
       shl cx, 1
       jmp L
Yes:
       mov eax, 0
       jmp En
```

```
No:
       mov eax, 1
En:
       pop edx
       pop ecx
       pop ebx
       pop ebp
       ret 4
ArrSum endp

Start:
       push offset X                    ) НЕ понято условие "получает число X"
                                          НЕ "адрес числа X"
       call ArrSum
       outint eax
       exit
end Start
```

ОЦЕНКА=2

3. Написать макроопределение с заголовком

NumOddNotWord **macro** X:VarArg

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов m8, m16, m132 или m64. Макрос печатает число нечётных переменных из списка X, не являющихся словами (m16), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

```
NumOddNotWord macro X:VarArg
       mov ebx,0
for i ,<X>
       local L,M
  ifnb <i>                              1) Не понято условие
                                           " для пустого списка выдавать ответ –1" ifb <X>
       elif type i EQ 1                 2) elif ??????
       mov al,i
       cbw                              3) УЧИТЬ определять чётность БЕЗ ДЕЛЕНИЯ
                                        4) Почему ЗНАКОВЫЕ ? Нет в условии
       mov cl,2
       idiv cl
       cmp ah,1
       je M
       elif type i EQ 4                 2) elif ??????     УЧИТЬ
       mov eax,i
       cdq
       mov ecx,2
       idiv ecx
       cmp edx,1
       je M
elif type i EQ 8                        2) elif ??????
       mov eax,i
       cqo                              5) НЕТ ТАКОГО в нашей ЭВМ !
       mov ecx,2
       idiv ecx                         6) do : dq !!!
       cmp edx,1
       je M
       jmp L

M: add ebx,1
L : endif                               7) НЕЛЬЗЯ
 endm
outint ebx
endm
```

ОЦЕНКА=0

4. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

```
include console.inc
.data
       extrn Perem_jmp: dword
.code
Start:
       mov eax, Perem_jmp
       sub edx, edx
       mov ebx, 3
       div ebx
       cmp edx, 0
       je M
L:     jmp L
M:     jmp Perem_jmp
N:     exit                              1) ???
       end Start
```

ОЦЕНКА=6

---

5. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

CF:=OF **or** SF

```
    jno L; OF = 0
    jmp M
    jns L; SF = 0
M:
    stc; CF = 1
    jmp N
L:
    clc; CF=0
N:
```

ОЦЕНКА=6

108-№6. Ибрагимова С.В.

1. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

2. Написать макроопределение с заголовком

```
NumOddNotWord macro X:VarArg
```

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов m8, m16, m132 или m64. Макрос печатает число нечётных переменных из списка X, не являющихся словами (m16), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

3. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

4. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

```
CF:=OF or SF
```

1.

```
include console.inc
.data
A dd ?
.code
bits proc
;input offset X                1) ??? НЕТ, САМ X
;output  bl = 0 или 1
  push ebp
  mov ebp, esp
  push ebx
  push eax                    2) НЕЛЬЗЯ! Учить ФУНКЦИИ
  push ecx
  mov eax, [ebp + 8] ; парамер dd удобно передавать по значению
  shr eax, 1.              ; сдвиг вправо
  jc @outproc1          ; если флаг CF = 1
  xor bl, bl                ; любимый счётчик
  mov ecx, 31
@l:
  shr eax, 1               ; опять сдвиг вправо
  jnc @skipinc          ; если флаг CF = 0
  inc bl
  jmp @skipxor
@skipinc:
  xor bl, bl
@skipxor:
  cmp bl, 3               ; есть ли 3 кол-ва единиц
  je @outproc1         ; если равно
  loop @l
  mov bl, 0
  jmp @out
@outproc1:
  mov bl, 1                   2) НЕТ! ФУНКЦИИ работают не так
  @out:
  pop ecx
  pop eax
  pop ebx
  pop ebp
  ret 4*1
bits endp
```

```
    start:
      inint A
      push A
      call bits
      outintln bl
      exit
    end start
```

ОЦЕНКА=0

2.

Без контроля ввода параметров

```
    NumOddNotWord macro X:  VarArg
ifnb <x>                                        1) X <> x !!!
            push ebx
            xor ebx, ebx     ; любимый счётчик
    for P, <x>
            local skip1, skip2                  2) Можно skip1=skip2
      if type P ne word
        if type P eq qword
            test dword ptr P, 1b  ; проверка
            jz skip1            ; если флаг ZF =1
            inc ebx
    skip1:
        else
            test P, 1b      ;  проверка
            jz skip2        ; если флаг ZF =1
            inc ebx
    skip2:
        endif
      endif
    endm
            outintln ebx
            pop ebx
else
            outint -1
endif
    endm
```

ОЦЕНКА=5

3.

```
include console.inc
.code
      extern Perem_jmp: dword
Start:
      mov eax, Perem_jmp
      mov edx, 0 /            1) / ???
      mov ebx, 3
      div ebx
      cmp edx, 0; если остаток равен не нулю то прыгаем
```

```
      jne looop
      mov eax, Perem_jmp                        2) Плохой переход, надо
                                                   jmp Perem_jmp
      mov eax, [eax]
      jmp eax
looop:  jmp looop       ; зацикливаемся
      ex                                        3) ???
      end Start
```

ОЦЕНКА=4

4.

```
      push eax
      jo one    ; если OF = 1
      js one    ;  если SF = 1
; CF = 0
      mov eax, 0
      rcr eax, 1  ; циклический сдвиг
      jmp fin
one:
      mov eax, 1
      rcr eax, 1
fin:
      pop eax
```

ОЦЕНКА=60

108-№8. Какунин К.В.

1.Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 10x20 *вещественных* чисел и выводит сумму номеров тех строк этой матрицы, которые начинаются с 0.0. При записи кодов операций использовать мнемонические обозначения.

```
001  ВВВ 100 200 000
002  ВЧВ 000 100 011  проверка на 0.0
003  ПНР 000 005 000
004  СЛЦ 014 014 012  + в сумму
005  СЛЦ 012 012 013  некст строчка
006  СЛЦ 002 002 016  первый элемент строчки
007  ВЧЦ 000 002 015  конец ли?
008  ПНР 000 002 000
009  ВЫВ 014 001 000  принт суммы
010 СТОП 000 000 000
011   00 000 000 000  0.0
012   00 000 000 001  N=1
013   00 000 000 001  +1
014   00 000 000 000  Сумма
015  ВЧВ 000 300 011  Проверка конца
016  00 000 020 000   Переход новой строки
```

ОЦЕНКА=6

---

2. Написать макроопределение с заголовком

```
NumOddNotWord macro X:VarArg
```

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов `m8`, `m16`, `m132` или `m64`. Макрос печатает число нечётных переменных из списка X, не являющихся словами (`m16`), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

```
NumOddNotWord macro X:VarArg
      local flager            0) УЧИТЬ ifb <X>
      flager = 0              1) Зачем, если ниже сразу flager = 1 ?
      push EAX
      mov EAX, 0
for arg, <X>
      local M
      flager = 1              1) ???
  if type arg EQ 1
      test arg, 1             2) СОВПАДАЮТ ???
      jz M                    2) СОВПАДАЮТ ???
      inc EAX                 2) СОВПАДАЮТ ???
M:
  endif
;132 рассматриваем как 32     3) ???
  if type arg EQ 4
      test arg, 1             2) СОВПАДАЮТ ???
      jz M                    2) СОВПАДАЮТ ???
      inc EAX                 2) СОВПАДАЮТ ???
M:
  endif
  if type arg EQ 8
      test dword ptr arg, 1
      jz M
      inc EAX
M:
  endif
endm
      outwordln EAX                      4) Надо УСЛОВНУЮ печать
      pop EAX
if flager EQ 0; если список пустой       5) ПОЗДНО, УЖЕ outwordln EAX
      outwordln -1
endif
      endm
```

ОЦЕНКА=3

3. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

```
include console.inc
.data
.code
Start:
      extrn Perem_jmp:dword
      mov EDX, 0
      mov EAX, Perem_jmp
      mov EBX, 3
      div EBX
      cmp EDX, 0
      jnz Start
      jmp Perem_jmp; косвенный переход
      exit                                    1) ?????
      end Start;
```

ОЦЕНКА=5

4. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

```
            CF:=OF or SF

      jno L1; если OF := 0                1) НЕВЕРНО! Это
                                             if (OF=0) and (SF=1) then CF:=1
      stc
L1:   jns L2; если OF := 0 и SF :=0
      stc
      jmp L3
L2: clc
L3:
```

ОЦЕНКА=0

108-№16. Сахаутдинова А.Р.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 10x20 *вещественных* чисел и выводит сумму номеров тех строк этой матрицы, которые начинаются с 0.0. При записи кодов операций использовать мнемонические обозначения.

2. Написать макроопределение с заголовком

```
NumOddNotWord macro X:VarArg
```

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов `m8`, `m16`, `m132` или `m64`. Макрос печатает число нечётных переменных из списка X, не являющихся словами (`m16`), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

3. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем `Perem_jmp` размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

4. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

```
CF:=OF or SF
```

№2.

```
NumOddNotWord macro              1) X:VarArg !!! ЗДЕСЬ !!!
X: VarArg
        local count
        count = 0
ifb <X>
   outlintln -1
else
  for p, <X>
    if type p NE 2
        local L
        test 1, byte ptr p       2) НЕТ такой команды !
        jz L
        count = count + 1        3) Эта ДИРЕКТИВА будет работать ВСЕГДА!
L:
     endif
  endm
        outlintln count
endif
endm
```

ОЦЕНКА=0

№3.

```
include console.inc
        extern Perem_jmp: dword
.data
.code
Start:
        mov eax, Perem_jmp
        mov ebx, 3
        xor edx, edx
        div ebx
L:      cmp edx, 0
        jNE L
        mov eax, Perem_jmp
        jmp dword ptr [eax]          1) НЕВЕРНО, это
                                        goto Perem_jmp↑↑, а надо
                                        goto Perem_jmp↑
                                     1) Просто jmp Perem_jmp
        pause                        2) ???
        exit                         2) ??? НИКОГДА не работают !
        end Start
```

ОЦЕНКА=3

№4.

```
   stc
   jo F
   js F
   clc
F:
```

ОЦЕНКА=6

№1.

```
001—ВВВ— 100—200—000
002—ВЧВ— 000—100—011
003—ПНР— 000—005—000
004—СЛЦ— 015—015—014
005—СЛЦ— 014—014—016
006—СЛЦ— 002—002—012
007—ВЧЦ— 000—014—013
008—ПНР— 000—002—000
009—ВЫВ— 015—001—000
010—СТОП—000—000—000
011—  00—000—000—000; константа 0                        1) 0.0
012—  00—000—020—000; переход на начало след стр
013—  00—000—000—010; константа 10
014—  00—000—000—001; номер строки
015—  00—000—000—000; сумма
016—  00—000—000—001; константа 1
```

ОЦЕНКА=6

108-№15. Савицкий И.П.

1. Написать программу в кодах учебной машины УМ-3, которая сначала вводит матрицу 10x20 ***вещественных*** чисел и выводит сумму номеров тех строк этой матрицы, которые начинаются с 0.0. При записи кодов операций использовать мнемонические обозначения.

001	ВВВ	100	200	000		
002	ВЧВ	000	100	014		
003	ПНР	000	005	000		
004	СЛЦ	012	012	011		
005	СЛЦ	011	011	013		
006	СЛЦ	002	002	015		
007	ВЧЦ	000	002	016		
008	ПНР	000	002	000		
009	ВЫВ	012	001	000		
010	СТОП	000	000	000		
011		000	000	001	i	1) НЕТ, 00 000 000 001
012		000	000	000	S	1) НЕТ, 00 000 000 000
013		000	000	001	Const 1	1) НЕТ, 00 000 000 001
014		000	000	000	Const 0.0	1) НЕТ, 00 000 000 000
015		000	020	000		1) НЕТ, 00 000 0й0 001
016	ВЧВ	000	300	014		Команда которую мы вычитаем для сравнения

ОЦЕНКА=3

---

2. Написать полную программу на Ассемблере, которая последовательно вводит (по **inint**) целые знаковые числа, пока не будет введёно число № (признак конца ввода). Программа выводит сумму тех из этих чисел, которые чётные и без остатка делятся на №. Если сумма превышает `MaxLongInt`, то вывести "Большая сумма".
СПЕЦИФИКАЦИЯ: последний ввод (а именно, число 15) учитывается в последовательности и всегда в нее входит
СПЕЦИФИКАЦИЯ: как только переполняется переменная, программа дальше не считает, а прекращает ввод и выводит "Большая сумма"

```
    .data
       Sum dd 0
    .code
    Start:
       mov bx, 15
    L:
       inint ecx          ; ввели знаковое число
       test ecx, 1        ; проверили на четность
       jnz  continue   ; нечетное — отправили дальше
       mov  eax, ecx ; готовимся к делению с сохранением числа
       shr  eax, 16
       mov  dx, ax
       mov  eax, ecx
       idiv bx            ; делим 4 байта(dx:ax) на 2(bx)

                    1) НАДО ДЛИННОЕ деление 8 байт(edx:eax) на 4 (ebx)
                       Понять ОБЯЗАТЕЛЬНО


       cmp dx, 0      ; сравниваем остаток с 0
       jne continue  ; не 0 — идем дальше
       add Sum, ecx; прибавляем наше число
       jo overflow    ; если случилось знаковое(поскольку maxlongint) переполнение, то
выводим диагностику и                          ; завершаем программу
continue:
       cmp ecx, 15  ; сравниваем, а не пора ли закругляться?
       jne L              ; если не равно — не пора
       outword Sum ;выводим суссу если все хорошо
```

```
                    2) НЕТ, outint Sum

      jmp ending   ; переходим на конец
   overflow:
      outstr „Большая сумма“; тут должны быть две верхнии кавычки, но мой редактор
ставит разные
   ending:
      exit
   end Start
```

ОЦЕНКА=2

---

3. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.
СПЕЦИФИКАЦИЯ: результат на al потому что помещается в 8 бит

```
func proc
      push ebp
      mov  ebp, esp
      push ebx
      push ecx
      push edx
      mov  edx, [ebp + 8]
      mov  al, 1
      test edx, 1
      jnz  ending
      mov  ecx, 30; 32-2, так как с шириной в три нам надо сделать на 2 шага меньше
L:
      test edx, 1
      jz   continue
      test edx, 010b
      jz   continue
      test edx, 0100b ; проверить сразу три единицы при помощи(test  edx, 111) не
получится, так как его нормально не сравнить (вдруг там 101), поэтому я посчитал
решение «в лоб» эффективнее, чем маскировать, потом двигать shr вправо и сравнивать с
0111b
      jz  continue
      mov al, 0
      jmp ending
continue:
      shr edx, 1
      loop L
ending:
      pop edx
      pop ecx
      pop ebx
      pop ebp
      ret 4
func endp

Start:
      push A; где A — то, что мы проверяем
      call func
      ; тут на al результат
end Start
```

ОЦЕНКА=6

---

4. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

```
          CF:=OF or SF
      clc; CF:=0
      jo true
      js true
      jmp ending
true:
      cmc; инвертирование CF
ending:
```

ОЦЕНКА=6

108-№19. Толеутаева А.Б.

1. Написать полную программу на Ассемблере, которая последовательно вводит (по **inint**) целые знаковые числа, пока не будет введёно число № (признак конца ввода). Программа выводит сумму тех из этих чисел, которые чётные и без остатка делятся на №. Если сумма превышает MaxLongInt, то вывести "Большая сумма".

2. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

3. Написать макроопределение с заголовком

NumOddNotWord **macro** X:VarArg

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов m8, m16, m132 или m64. Макрос печатает число нечётных переменных из списка X, не являющихся словами (m16), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

4. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

1. N = 19

```
.686
include console.inc
.data
   T dd ?
.code
Start:
   Sub ecx, ecx ;summ
L:
   Inint T ;целое знаковое
   Jc  Bad_arg
   Cmp T, 19 ;не считаем, это служебный символ
   Je  next
   Test T, 1
   Jnz L ;Нечетное
   cmp T, 0
   jl  Min
;проверка на кратность 19
   xor edx, edx
   Mov eax, T
   Mov ebx, 19
   Div ebx
   Cmp edx, 0
   Jne L
   Add ecx, T
   Jo Big_sum
   Jmp L
Min:
   neg T                        1) Существует neg T ??
                                   Надо ЗНАКОВОЕ деление
   xor edx, edx
   Mov eax, T
   Mov ebx, 19
   Div ebx
   Cmp edx, 0
   Jne L
   sub ecx, T
   Jo Big_sum
   Jmp L
Big_sum:
   outstr 'Большая сумма'
   exit
Bad_arg:
   outstr 'Bad variable'
   exit
next:
   outintln ecx
```

```
    exit
end Start
```

ОЦЕНКА=5

---

2. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

19 mod 3 = 1. Младший бит 0 -> четное

```
T dw ?
```

Вызов:

```
   Movsx eax, T
```

1) НЕТ, надо `movzx`, иначе будет много "1" подряд !

```
; использую movsx, потому что movzx заполнила бы пустоту знаковым битом, и ответ всегда выводился бы 0
; для четных чисел и 1 для нечетных.
   Push eax
   Call P
   Outint eax

P proc
   push ebp
   mov  ebp, esp
   push ebx
   push ecx
   push edx
   sub  dl, dl ;кол-во единиц
   mov  eax, 1
   mov  ebx, [ebp + 8]; T
   test ebx, 1
   jnz L ;младший бит не ноль
testing:
   shr ebx, 1
   jc L1 ;в конце единица
   sub dl, dl ;Обнуляем счетчик единиц
   jmp next
L1:inc dl
   cmp dl, 3
   je change_answer
next:
   cmp ebx, 0
   jne testing
change_answer:
   sub eax, eax
L:
   Pop edx
   pop ecx
   pop ebx
   Pop ebp
   ret 4
   P endp
```

ОЦЕНКА=4

---

4. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

include console.inc

```
.data
      extern Perem_jmp: dword
.code
begin:
      mov eax, Perem_jmp
      mov ecx, 3
      xor edx,edx
      div ecx
```

```
      cmp edx, 0
      jne begin
      mov eax,Perem_jmp
      jmp [eax]                    1) НЕТ, надо goto Perem_jmp↑
                                      а это goto Perem_jmp↑↑
                                      Просто jmp Perem_jmp

      exit                         2) Никогда не работает
      end begin
```

ОЦЕНКА=3

---

3. Написать макроопределение с заголовком

NumOddNotWord **macro** X:VarArg

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов m8, m16, m132 или m64. Макрос печатает число нечётных переменных из списка X, не являющихся словами (m16), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

```
NumOddNotWord macro X: VarArg
ifb <X>
            mov eax,-1
else
            mov eax,0
  for i,<X>
            local L
    if type i NE 2
            test i,1
            jz L
            inc eax
L:
    endif
  endm
endif
            outint eax
      endm
```

ОЦЕНКА=6

108-№20. Тунис И.С.

1. Написать полную программу на Ассемблере, которая последовательно вводит (по **inint**) целые знаковые числа, пока не будет введёно число № (признак конца ввода). Программа выводит сумму тех из этих чисел, которые чётные и без остатка делятся на №. Если сумма превышает `MaxLongInt`, то вывести "Большая сумма".

2. Написать макроопределение с заголовком

```
NumOddNotWord macro X:VarArg
```

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов `m8`, `m16`, `m132` или `m64`. Макрос печатает число нечётных переменных из списка X, не являющихся словами (`m16`), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

3. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

```
CF:=OF or SF
```

Тунис Илья третья итерация

1.

```
include console.inc
.code
Begin:
        mov ebx, 20
        mov ecx, 0
        mov bp, 0
cikl:
        inint esi
        cmp esi, 20
        je  fin
        mov eax, esi
        cdq ; конвертация edx:eax := eax
        idiv ebx
        cmp edx,0
        jne nott
        add ecx, esi
                        1) НЕТ проверки переполнения суммы !
                           JO ERROR
        mov bp,1
nott:
        jmp cikl
fin:
        cmp bp,0
        jne nenol
        outint ecx
        jmp eend
nenol:
        outstr 'Большая сумма'
eend:
        exit
        end Begin
```

ОЦЕНКА=3

№2

```
NumOddNotWord macro X: vararg
ifb <X>
        outstr '-1'
else
        push EAX
        xor EAX, EAX
  for T, <X>
        local notInc
    if type T NE 2
       if type T EQ 8
        test dword ptr T, 1
        jz notInc
```

```
            inc EAX
notInc:
      else
            test T, 1
            jz notInc
            inc EAX
notInc:
      endif
    endif
  endm
      outintln EAX
      pop EAX
endif
      endm
```

ОЦЕНКА=6

3.

ОЦЕНКА=0

108-№5. Жуков Н.А.

1. Написать на Ассемблере функцию со стандартными соглашениями о связях, которая получает в качестве параметра целое число X в формате **T** (T=**dd** для № **mod** 3=0; T=**dw** для № **mod** 3=1; T=**db** для № **mod** 3=2). Функция вырабатывает 0, если X имеет младший бит "0" и в X есть хотя бы три идущих подряд бита "1", иначе функция вырабатывает 1. Привести пример вызова этой функции.

2. Написать макроопределение с заголовком

```
NumOddNotWord macro X:VarArg
```

X – это список $X_1,X_2,\ldots,X_k$, где X_i – имена переменных форматов m8, m16, m132 или m64. Макрос печатает число нечётных переменных из списка X, не являющихся словами (m16), для пустого списка выдавать ответ –1. Контроль правильности типов параметров не проводить.

3. Написать на Ассемблере головной модуль, который делает переход **jmp** по адресу, содержащемуся в беззнаковой переменной с именем Perem_jmp размером в двойное слово (**dd**), описанной в каком-то другом модуле, если эта переменная кратна 3, иначе зацикливается.

4. Считая фраги логическими переменными, написать фрагмент на Ассемблере для выполнения действия:

```
CF:=OF or SF
```

№1

```
Include console.inc
.data
x db ?
.code
yur proc
;[ebp+8] - x:byte
      push ebp
      mov ebp,esp
      push ecx
      push ebx
      mov eax,1
      mov bl,[ebp+8]
      shr bl,1
      jc fin
      mov ecx,5
cyc:
      mov bh,bl
      and bl,7
      cmp bl,7
      je zer
      mov bl,bh
      shr bl,1
      loop cyc
      jmp fin
zer:
      xor eax,eax
fin:
      pop ebx
      pop ecx
      pop ebp
      ret 4
yur endp

start:
      movzx eax,x
      push eax
      call yur
      exit
end start
```

ОЦЕНКА=6

№2

```
include console.inc
Numnotoddword macro X :varArg
ifnb <X>
            push eax
            push ebx
```

```
            xor ebx, ebx
  for Q, <X>
    if type Q NE 2
      if type Q EQ 8
            mov eax, Q + 4          1) dword ptr Q
            shr eax, 1              2) mov eax, dword ptr Q
                                       Перевернутое в памяти
            adc ebx, 0
      endif
      if type Q EQ 4
            mov eax, Q              3) ВСЮДУ просто test Q,1 !!!
            shr eax, 1
            adc ebx, 0;
      endif
      if type Q EQ 1
            mov al, Q
            shr al, 1
            adc ebx, 0
      endif
    endif
  endm; for
            outword ebx
            pop ebx
            pop eax
else
            echo "-1"               4) outint -1; ПОНЯТЬ ИЛИ СПРОСИТЬ
            exitm
endif
endm
```

ОЦЕНКА=4

№3

```
include console.inc
      extern perem_jmp:dword
.code tart:
      xor edx,edx
      mov eax,perem_jmp
      mov ecx,3
      div ecx
      cmp edx,0
      jne fin
      mov eax,perem_jmp
      jmp eax
fin:                          1) Не понято условие задачи "Зацклиться"
end start
```

ОЦЕНКА=2

№4

```
      mov al,1
      js  tr
      jno fa
tr:   shr al,1
      jmp cnt
fa:   shl al,1
cnt:
```

ОЦЕНКА=6
