**Práctica de Examen**

**Nombre:** Vargas Cruz Jose Manuel **Carrera:** Ing. Informática

**Ejercicio 1:**

```
org 100h


inicio:
   mov cx,0
   mov ah,1
   mov si,offset cad1
   mov di,offset cad2

ciclo:
   int 21h

   cmp al,13
   je ciclo3
   mov bl,al
   push bx
   inc cl
   jmp ciclo

ciclo3:
   mov ax,0
   mov ax,cx
   push ax
   pop bx
   jmp ciclo2

ciclo2:
```

```
    pop bx
    cmp bl,'A'
    jb mincar

    cmp bl,'Z'
    ja addmincar
    cmp bl,40h
    ja numay

    cmp bl,39h
    jb numay2
    tag:

    loop ciclo2
    push ax



    pop ax
    mov cx,ax

    mov si,offset cad1
    mov di,offset cad2
    mov ah,2


printCads:
    cmp cx,0
    je fin
    call salto_linea
```

```
    cmp [si],0
    je continue1
    jne printSi


printSi:
    mov dl,[si]
    int 21h
    sub cx,1
    jmp continue
continue1:
    mov dl,' '
    int 21h
    jmp continue
continue:

    inc si
    cmp [di],0
    je cont
    jne printDi
cont:
    mov dl,' '
    int 21h
    jmp printCads
printDi:

    mov dl,[di]
    int 21h
    inc di
```

```
    sub cx,1
    jmp printCads




    jmp fin



mincar:

    cmp bl,5Ah
    ja addmincar
    cmp bl,39h
    ja addmincar
    cmp bl,30h
    jb addmincar

    cmp bl,30h
    ja mov numay2

numay:
    cmp bl,'Z'
    jb addnumay
    inc di

numay2:
    cmp bl,30h
```

```
    ja addnumay
    inc di


addmincar:
    mov [si],bl
    inc si
    jmp tag
addnumay:
    mov [di],bl
    inc di
    jmp tag


fin: int 20h
proc salto_linea
    mov ah,2
    mov dl,10
    int 21h
    mov dl,13
    int 21h

    mov dx,0
    ret
salto_linea endp

cad1 db 50 dup (0)
cad2 db 50 dup (0)
```

## Ejercicio 2

```
saveNum macro dig
   local for
   push ax
   push cx
   push dx
   mov cx,dig
   mov bx,10
   mov dx,0
   push dx
   for:
      mov ah,7
      int 21h
      cmp al,'0'
      jb  for
      cmp al,'9'
      ja  for
      mov dl,al
      mov ah,2
      int 21h
      pop ax
      sub dl,30h
      mov dh,0
      push dx
      mov dx,0
      mul bx
      pop dx
      add ax,dx
      push ax
```

```
        loop for
    pop bx
    pop dx
    pop cx
    pop ax
endm
printMsg macro msg
    push ax
    push dx
    mov dx,offset msg
    mov ah,9
    int 21h
    pop dx
    pop ax
endm

printNum macro reg,base
    local while,break,for,show
    push ax
    push bx
    push cx
    push dx
    mov ax,reg
    mov bx,base
    mov cx,1
    while:
        cmp ax,bx
        jb  break
        mov dx,0
```

```
        div bx
        push dx
        inc cx
        jmp while
    break:
    push ax
    mov ah,2
    for:
        pop dx
        add dl,30h
        cmp dl,'9'
        jbe show
        add dl,7h
        show:
        int 21h
        loop for
    pop dx
    pop cx
    pop bx
    pop ax
endm



printChar macro char
    push ax
    push dx
    mov ah,2
    mov dl,char
```

```
    int 21h
    pop dx
    pop ax
endm

org 100h
inicio:
    printMsg msg
    printMsg date
    printMsg form
    printChar 13
    printMsg date
    saveNum 2
    printChar '-'
    mov cx,0
    mov cl,bl
    mov ax,bx
    saveNum 2
    printChar '-'
    mov dx,0
    mov dl,bl
    shl bx,5
    or  ax,bx
    saveNum 4
    push bx
    sub bx,1950
    shl bx,9
    or  ax,bx
    pop bx
```

```
    printMsg bin
    printMsg date
    printNum ax,2
    printMsg year
    printNum bx,2
    printMsg month
    printNum dx,2
    printMsg day
    printNum cx,2
    printMsg hex
    printMsg date
    printNum ax,16
    printMsg year
    printNum bx,16
    printMsg month
    printNum dx,16
    printMsg day
    printNum cx,16

finish:
    int 20h

msg db 'Ingrese la fecha en formato DD-MM-AAAA',10,13,24h
day db 10,13,'Dia: $'
month db 10,13,'Mes: $'
year db 10,13,'A¤o: $'
date db 'Fecha: $'
form db '__-__-____$'
bin db 10,10,13,'Binario:',10,13,'$'
```

```
hex db 10,10,13,'Hexadecimal:',10,13,'$'
```

---

### Ejercicio 3

```
printInBase macro reg,base
   local digits, minor,num,show
   push ax
   push bx
   push cx
   push dx
      mov al,reg
      mov bl,base
      mov ah,0
      mov cx,1
      mov dx,0
      digits:
         cmp al,bl
         jb  minor
         mov bh,0
         div bl
         mov dl,ah
         mov ah,0
         push dx
         inc cx
         jmp digits
   minor:
      push ax
      mov ah,2
      show:
         pop dx
         cmp dl,10
```

```
        jb  num
        add dl,7
      num:
        add dl,30h
        int 21h
        loop show
   pop dx
   pop cx
   pop bx
   pop ax
endm
capture2 macro reg
   push ax
   call capture1
   mov reg,al
   shl reg,4
   call capture1
   or  reg,al
   pop ax
endm

printChar macro char
   push ax
   push dx
   mov ah,2
   mov dl,char
   int 21h
   pop dx
   pop ax
```

```
endm

sortAsc macro reg1,reg2
   local fin
   cmp reg1,reg2
   jb  fin
   xchg reg1,reg2
   fin:
endm



printVar macro var
   push ax
   push dx
      mov ah,9
      mov dx,offset var
      int 21h
   pop dx
   pop ax
endm

org 100h
begin:
   mov bh,0
   mov ch,0
   printVar msg
   capture2 bl
   printChar ' '
```

```
    capture2 cl
    printChar ' '
    capture2 dl
    printChar ' '
    ;sortAsc bl,cl
    ;sortAsc bl,dl
    ;sortAsc cl,dl
    cmp bl,cl
    jb  fin1
    xchg bl,cl
fin1:
    cmp bl,dl
    jb  fin2
    xchg bl,dl
fin2:
    cmp cl,dl
    jb  fin3
    xchg cl,dl
fin3:
    push bx
    push cx
    printVar may
    printInBase dl,2
    pop dx
    printVar mid
    printInBase dl,10
    pop dx
    printVar min
    printInBase dl,8
```

```
finish:
   int 20h


capture1 proc
   push dx
   while1:
   mov ah,7
   int 21h
   cmp al,'0'
   jb  while1
   cmp al,'9'
   jbe pass
   cmp al,'A'
   jb  while1
   cmp al,'F'
   ja  while1
   pass:
   mov dl,al
   mov ah,2
   int 21h
   pop dx
   sub al,30h
   cmp al,10
   jb  endProc
   sub al,7
endProc:
   ret
capture1 endp
```

```
msg db 'Ingrese tres numeros hexadecimales de dos digitos: $'
may db 10,13,'Mayor: $'
mid db 10,13,'Medio: $'
min db 10,13,'Menor: $'
```

---

### Ejercicio 4:

```
printValue macro num
   local bucle1,bucle2,break
   push ax
   push bx
   push cx
   push dx
   mov ax,num
   mov cx,1
   mov bx,10
   bucle1:
      cmp ax,bx
      jb  break
      mov dx,0
      div bx
      push dx
      inc cx
      jmp bucle1
break:
   push ax
   mov ah,2
   bucle2:
      pop dx
      add dx,30h
```

```
        int 21h
        loop bucle2
    pop dx
    pop cx
    pop bx
    pop ax
endm

printVarValue macro var,lim
    local bucleP,finPrint
    push ax
    mov si,offset var
    bucleP:
        mov ax,[si]
        cmp ax,lim
        je  finPrint
        printValue ax
        printChar 10
        printChar 13
        add si,2
        jmp bucleP
    finPrint:
    pop ax
endm

printChar macro char
    push ax
    push dx
    mov ah,2
```

```
    mov dl,char
    int 21h
    pop dx
    pop ax
endm

        org 100h
        begin:
            mov cx,50
            mov si,offset par
            mov di,offset npr
            mov bp,offset pri
            bucle:
                call saveNum
                cmp dl,1
                je  enter
                call isPrime
                cmp dl,0
                je  pair1
                mov [bp],ax
                add bp,2
                jmp for1
            pair1:
                call isImpair
                cmp dl,1
                je impair1
                mov [si],ax
                add si,2
                jmp for1
```

```
            impair1:
               mov [di],ax
               add di,2
               for1:
               loop bucle
            enter:
               printVarValue pri,1000
               printVarValue par,1000
               printVarValue npr,1000

         finish:
            int 20h

; num is ax
; cx is lim
isPrime proc
   push bx
   push cx
   cmp ax,2
   jb  nonprime
   mov bx,2
   mov cx,ax
   bucle1:
      push ax
      cmp bx,cx
      jnb prime
      mov dx,0
      div bx
      cmp dx,0
```

```
        je  nonprime
        mov cx,ax
        inc bx
        pop ax
        jmp bucle1
    prime:
        mov dl,1
        jmp fin
    nonprime:
        mov dl,0
        jmp fin
    fin:
    pop ax
    pop cx
    pop bx
    ret
isPrime endp

;guarda en ax
saveNum proc
    push bx
    push cx
    mov cx,3
    mov bx,10
    mov dx,0
    push dx
    bucle2:
        mov ah,7
        int 21h
```

```
        cmp al,13
        je  enter2
        cmp al,'0'
        jb  bucle2
        cmp al,'9'
        ja  bucle2
        mov dl,al
        mov ah,2
        int 21h
        sub dl,30h
        pop ax
        push dx
        mov dx,0
        mul bx
        pop dx
        add ax,dx
        push ax
        loop bucle2
enter2:
    mov dx,0
    printChar 13
    printChar 10
    cmp cx,3
    jne pass
    mov dl,1
    pass:
    pop ax
    pop cx
    pop bx
```

```
    ret
saveNum endp

isImpair proc
    push ax
    mov dl,1
    shr ax,1
    jc  impar
    mov dl,0
    impar:
    pop ax
    ret
isImpair endp

bool proc
    push ax
    push dx
    mov ah,9
    cmp dl,1
    je true
    mov dx,offset f
    jmp done
    true:
    mov dx,offset t
    done:
    int 21h
    pop dx
    pop ax
    ret
```

```
bool endp

par dw 50 dup(1000)
npr dw 50 dup(1000)
pri dw 51 dup(1000)
t db 'true',10,13,'$'
f db 'false',10,13,'$'
```

---

**Ejercicio 5:**

```
printStr macro var
   push ax
   push dx
   mov ah,9
   mov dx,offset var
   int 21h
   pop dx
   pop ax
endm

printChar macro char
   push ax
   push dx
   mov dl,char
   mov ah,2
   int 21h
   pop dx
   pop ax
endm
```

```
printNum macro reg,base
   local while,break,for,pass
   push ax
   push bx
   push cx
   push dx
   mov ax,reg
   mov bx,base
   mov cx,1
   while:
      cmp ax,bx
      jb  break
      mov dx,0
      div bx
      push dx
      inc cx
      jmp while
   break:
   push ax
   mov ah,2
   for:
      pop dx
      add dl,30h
      cmp dl,'9'
      jb  pass
      add dl,7h
      pass:
      int 21h
      loop for
```

```
    pop dx
    pop cx
    pop bx
    pop ax
endm
saveNum macro
    local while
    push ax
    while:
    mov ah,7
    int 21h
    cmp al,'0'
    jb while
    cmp al,'9'
    ja while
    mov ah,2
    mov dl,al
    int 21h
    sub dl,30h
    pop ax
endm

saveNum2dig macro var
    local for,break
    push ax
    push dx
    push cx
    push bx
    mov bl,10
```

```
    mov cx,2
    push 0
    for:
      mov ah,7
      int 21h
      cmp al,13
      je break
      cmp al,'0'
      jb for
      cmp al,'9'
      ja for
      mov dl,al
      mov ah,2
      int 21h
      sub dl,30h
      pop ax
      mul bl
      add al,dl
      push ax
      loop for
    break:
      pop ax
      mov var,al
    pop bx
    pop cx
    pop dx
    pop ax
endm
```

```
saveOp macro var
    local verif,break
    push ax
    push dx
    verif:
    mov ah,7
    int 21h
    cmp al,'+'
    je break
    cmp al,'-'
    je break
    cmp al,'*'
    je break
    cmp al,'/'
    jne verif
    break:
    mov ah,2
    mov dl,al
    mov var,dl
    int 21h
    pop dx
    pop ax
endm



org 100h
begin:
```

```
printStr menu
saveNum
cmp dl,1
je  ingNum
cmp dl,2
je  ingOp
cmp dl,3
je  show
jmp begin
ingNum:
printStr num1
saveNum2dig n1
printStr num2
saveNum2dig n2
printChar 10
printChar 13
jmp begin
ingOp:
printStr opMsg
saveOp op
printChar 10
printChar 13
jmp begin
show:
mov dl,op
cmp dl,0
je  begin
mov ax,0
mov al,n1
```

```
mov bl,n2
cmp dl,'+'
je  sum
cmp dl,'-'
je  rest
cmp dl,'*'
je  por
div bl
mov ah,0
jmp show2
por:
mov dx,0
mul bx
jmp show2
sum:
add al,bl
jmp show2
rest:
cmp al,bl
jb  show3
sub al,bl
show2:
printStr res
printNum ax,2
printStr oct
printNum ax,8
printStr deci
printNum ax,10
printStr hex
```

```
    printNum ax,16
    jmp finish
    show3:
    sub bl,al
    mov bh,0
    printStr res
    printChar '-'
    printNum bx,2
    printStr oct
    printChar '-'
    printNum bx,8
    printStr deci
    printChar '-'
    printNum bx,10
    printStr hex
    printChar '-'
    printNum bx,16

finish:
    int 20h

menu db '1 Ingreso de numeros',10,13
     db '2 Ingreso de operacion',10,13
     db '3 Mostrar resultado',10,13,24h
num1  db 10,13,'Numero1: $'
num2  db 10,13,'Numero2: $'
opMsg   db 10,13,'Operador: $'
res  db 10,13,'Resultado: ',10,13
     db 'Binario: $'
```

```
oct  db 10,13,'Octal: $'
deci db 10,13,'Decimal: $'
hex  db 10,13,'Hexadecimal: $'
n1   db 0
n2   db 0
op   db 0
```

---

### Ejercicio 6

```
printStr macro var
   push ax
   push dx
   mov ah,9
   mov dx, offset var
   int 21h
   pop dx
   pop ax
endm

inputLetter macro
   local enter,while,pass
   push ax
   push dx
   while:
      mov ah,7
      int 21h
      cmp al,13
      je  enter
      cmp al,' '
      je  pass
```

```
        cmp al,'A'
        jb  while
        cmp al,'['
        jb  pass
        cmp al,'a'
        jb  while
        cmp al,'z'
        ja  while
        pass:
        mov ah,2
        mov dl,al
        int 21h
        jmp while
    enter:
    pop dx
    pop ax
endm

saveLetters macro var,lim
    local enter,for,pass
    push ax
    push dx
    push cx
    mov si, offset var
    mov cx,lim
    for:
        mov ah,7
        int 21h
        cmp al,13
```

```
        je  enter
        cmp al,' '
        je  pass
        cmp al,'a'
        jb  for
        cmp al,'z'
        ja  for
        pass:
        mov ah,2
        mov dl,al
        mov [si],dl
        inc si
        int 21h
        loop for
    enter:
    pop cx
    pop dx
    pop ax
endm

inputNum macro cnt
    local for
    push ax
    push cx
    push dx
    mov cx,cnt
    for:
        mov ah,7
        int 21h
```

```
        cmp al,'0'
        jb for
        cmp al,'9'
        ja for
        mov ah,2
        mov dl,al
        int 21h
        loop for
    pop dx
    pop cx
    pop ax
endm

inputNumDl macro
    local verif
    push ax
    push cx
    verif:
        mov ah,7
        int 21h
        cmp al,'0'
        jb verif
        cmp al,'9'
        ja verif
        mov ah,2
        mov dl,al
        int 21h
    pop ax
endm
```

```
telfCode macro
    local lp,sc,n4,n5,n6,n8,beni,potosi,chu,nn8,fin
    inputNumDl
    mov dh,0
    cmp dl,2
    je  lp
    cmp dl,3
    je sc
    cmp dl,4
    je n4
    cmp dl,5
    je n5
    cmp dl,6
    je n6
    cmp dl,8
    je n8
    inputNum 6
    jmp fin
    lp:
       inputNum 6h ;2xx xxxx
       mov dh,1
       jmp fin
    sc:
       inputNum 6h ;3xx xxxx
       mov dh,2
       jmp fin
    n4:
       inputNumDl
```

```
    inputNum 5h
    cmp dl,6
    je beni:
       mov dh,3    ;4xx xxxx
       jmp fin     ;cocha
    beni:
       mov dh,4    ;46x xxxx
       jmp fin
n5:
    inputNumDl
    inputNum 5h
    cmp dl,2     ;52x xxxx
    jne fin
    mov dh,5     ;oruro
    jmp fin
n6:
    inputNumDl
    inputNum 5h
    cmp dl,2
    je  potosi
    cmp dl,4
    je  chu
    cmp dl,6
    jne fin
       mov dh,6    ;tarija
       jmp fin     ;66x xxxx
    potosi:
       mov dh,7    ;62x xxxx
       jmp fin
```

```
      chu:
         mov dh,8    ;64x xxxx
         jmp fin
   n8:
      inputNumDl
      cmp dl,4
      je  next
      inputNum 5h
      jmp fin:
      next:
      inputNumDl
      inputNum 4h
      cmp dl,2       ;842 xxxx
      jne fin
      mov dh,9
   fin:
endm

compare macro var1,var2
   local while,break
   push bx
   mov si,offset var1
   mov di,offset var2
   mov dl,0
   while:
      mov bh,[si]
      mov bl,[si]
      inc si
      inc di
```

```
        cmp bh,bl
        jne break
        cmp bh,'$'
        jne while
        mov dl,1
    break:
    pop bx
endm

ifelse macro tag1,tag2
    cmp dl,1
    je  tag1
    jmp tag2
endm

org 100h
begin:
    printStr nom
    inputLetter
    printStr ape
    inputLetter
    printStr telf
    telfCode
    printStr ciu
    saveLetters city,10
    cmp dh,0
    je fail
    cmp dh,1
    jne next2
```

```
   compare city,lp
   ifelse pass,fail
next2:
cmp dh,2
jne next3
   compare city,sc
   ifelse pass,fail
next3:
cmp dh,3
jne next4
   compare city,co
   ifelse pass,fail
next4:
cmp dh,4
jne next5
   compare city,be
   ifelse pass,fail
next5:
cmp dh,5
jne next6
   compare city,oru
   ifelse pass,fail
next6:
cmp dh,6
jne next7
   compare city,ta
   ifelse pass,fail
next7:
cmp dh,7
```

```
    jne next8
        compare city,po
        ifelse pass,fail
    next8:
    cmp dh,8
    jne next9
        compare city,chu
        ifelse pass,fail
    next9:
        compare city,pa
        ifelse pass,fail
    pass:
        printStr reg
        jmp finish
    fail:
        printStr nreg
finish:
    int 20h

city db 11 dup('$')
nom db 'Nombre: $'
ape db 10,13,'Apellido(s): $'
telf db 10,13,'Telefono: $'
ciu db 10,13,'Ciudad: $'
reg db 10,13,'Registrado$'
nreg db 10,13,'Registro anulado$'
lp db 'la paz$'
sc db 'santa cruz$'
co db 'cochabamba$'
```

```
be db 'beni$'
oru db 'oruro$'
ta db 'tarija$'
po db 'potosi$'
chu db 'chuquisica$'
pa db 'pando$'
```

### Ejercicio 7

No funciona

---

### Ejercicio 8

No compila

---

### Ejercicio 9

No funciona

---

### Ejercicio 10

```
saveCantNum macro cant,var
   local for
   push ax
   push bx
   push cx
   push dx
   mov cx,cant
   for:
      mov ah,7
      int 21h
      cmp al,'0'
      jb  for
```

```
        cmp al,'9'
        ja  for
        mov dl,al
        mov ah,2
        int 21h
        mov bx,var
        inc bx
        mov var,bx
        loop for
    pop dx
    pop cx
    pop bx
    pop ax
endm

saveNum macro dig,var
    local for,break
    push ax
    push bx
    push cx
    push dx
    mov cx,dig
    for:
        mov ah,7
        int 21h
        cmp al,13
        je  break
        cmp al,'0'
        jb  for
```

```
        cmp al,'9'
        ja  for
        mov dl,al
        mov ah,2
        int 21h
        mov bx,var
        inc bx
        mov var,bx
        loop for
    break:
    pop dx
    pop cx
    pop bx
    pop ax

endm

saveLet macro cant,varC,varA
    local for,break,pass
    push ax
    push bx
    push cx
    push dx
    mov cx,cant
    for:
        mov ah,7
        int 21h
        cmp al,13
        je  break
```

```
        cmp al,' '
        je pass
        cmp al,'A'
        jb  for
        cmp al,'Z'
        ja  for
        mov ah,0
        add varA,ax
        inc varC
        pass:
        mov dl,al
        mov ah,2
        int 21h
        loop for
    break:
    pop dx
    pop cx
    pop bx
    pop ax
endm

printStr macro var
    push ax
    push dx
    mov ah,9
    mov dx,offset var
    int 21h
    pop dx
    pop ax
```

```
endm

printChar macro char
   push ax
   push dx
   mov ah,2
   mov dl,char
   int 21h
   pop dx
   pop ax
endm

printVal macro num
   local while,break,for
   push ax
   push bx
   push cx
   push dx
   mov ax,num
   mov bx,10
   mov cx,1
   while:
      cmp ax,bx
      jb  break
      mov dx,0
      div bx
      push dx
      inc cx
      jmp while
```

```
    break:
    push ax
    mov ah,2
    for:
        pop dx
        add dl,30h
        int 21h
        loop for
    pop dx
    pop cx
    pop bx
    pop ax

endm

org 100h
begin:
    printStr nit
    saveNum 10,cnt
    printStr nroFact
    saveNum 15,cnt
    printStr nroAut
    saveNum 15,cnt
    printStr nomCli
    saveLet 30,numLet,acum
    printStr fecha
    saveCantNum 2,cnt
    printChar '/'
    saveCantNum 2,cnt
```

```
    printChar '/'
    saveCantNum 4,cnt
    printStr cod
    mov ax,acum
    mov bx,numLet
    mov dx,0
    div bx
    printChar al
    mov ax,cnt
    add ax,bx
    printVal ax
finish:
    int 20h

nit     db 'NIT: $'
nroFact db 10,13,'Nø Factura: $'
nroAut  db 10,13,'Nø Autorizacion: $'
nomCli  db 10,13,'Nombre: $'
mont    db 10,13,'Monto: $'
fecha   db 10,13,'Fecha: $'
cod     db 10,10,13,'Codigo: $'
cnt     dw 0
numLet  dw 0
acum    dw 0
```