

## OBJETIVO

- Repasar paso a paso los principales conceptos teóricos de COBOL CICS a efectos que el estudiante pueda encarar su primer desafío de código COBOL pseudo-conversacional.

## ESPECIFICACIONES

### ¿QUÉ ES CICS?

CICS, que es el acrónimo de: *Customer Information Control System*, es un servidor de transacciones desarrollado por IBM que se utiliza principalmente en sistemas mainframe para gestionar aplicaciones críticas de negocio en tiempo real.

Fue creado en 1968 y desde entonces ha evolucionado para soportar tecnologías modernas como Java, Node.js etc.

Su función principal es permitir que múltiples transacciones se procesen de forma simultánea, garantizando integridad, consistencia y disponibilidad. Esto lo hace ideal para sectores como la banca, los seguros y el gobierno, donde se requiere alta disponibilidad y rendimiento.

CICS permite desarrollar aplicaciones en lenguajes como COBOL y otros más, lo que facilita la integración de sistemas antiguos con nuevas tecnologías.

Además, ofrece herramientas para monitoreo, seguridad avanzada y recuperación ante fallos.

En el sector bancario, **CICS se utiliza como la columna vertebral de muchas operaciones críticas**. Su capacidad para manejar miles de transacciones por segundo lo convierte en una herramienta ideal para servicios como:

- **Procesamiento de transacciones financieras:** depósitos, retiros, transferencias y pagos se ejecutan en tiempo real con alta confiabilidad.
- **Gestión de cuentas:** permite acceder y actualizar información de cuentas bancarias de forma segura.
- **Integración con cajeros automáticos (ATM) y banca en línea:** CICS actúa como intermediario entre los sistemas front-end y las bases de datos centrales.
- **Cumplimiento normativo y auditoría:** registra cada transacción con precisión, lo que facilita el seguimiento y la trazabilidad.
- **Seguridad y control de acceso:** protege los datos sensibles mediante autenticación robusta y control de permisos.

Además de la banca, **CICS se utiliza en una variedad de industrias que requieren procesamiento de transacciones de alto volumen y alta disponibilidad.** Algunos ejemplos son:

- **Seguros:** para gestionar pólizas, procesar reclamaciones y calcular primas en tiempo real.
- **Gobierno:** en sistemas de recaudación de impuestos, seguridad social y registros civiles, donde se necesita integridad de datos y trazabilidad.
- **Retail y comercio electrónico:** para manejar inventarios, ventas, devoluciones y programas de fidelización de clientes.
- **Salud:** en la administración de historiales médicos, facturación y autorizaciones de tratamientos.
- **Transporte y logística:** para reservas, seguimiento de envíos y gestión de flotas.

CICS también se adapta a entornos modernos gracias a su compatibilidad con Apis, servicios web y lenguajes como Java y Node.js. Esto permite que empresas con sistemas heredados puedan integrarse con nuevas plataformas sin rehacer todo desde cero.

Vamos a ver cómo **CICS se está modernizando con inteligencia artificial (IA)** para seguir siendo relevante en los entornos empresariales actuales.

IBM ha desarrollado formas de **infundir IA directamente en las aplicaciones que corren sobre CICS**, lo que permite tomar decisiones en tiempo real dentro de las transacciones.

Por ejemplo:

- **Evaluación de riesgos en préstamos:** una transacción puede consultar un modelo de IA para decidir si aprobar o no un crédito en el momento exacto en que el cliente lo solicita.
  - **Detección de fraude:** al procesar un reclamo de un seguro o una transacción bancaria, CICS puede invocar un modelo de IA que detecte patrones sospechosos sin salir del entorno mainframe.
  - **Ofertas personalizadas:** mientras un cliente interactúa con un canal digital, CICS puede usar IA para recomendar productos financieros adaptados a su perfil.
-

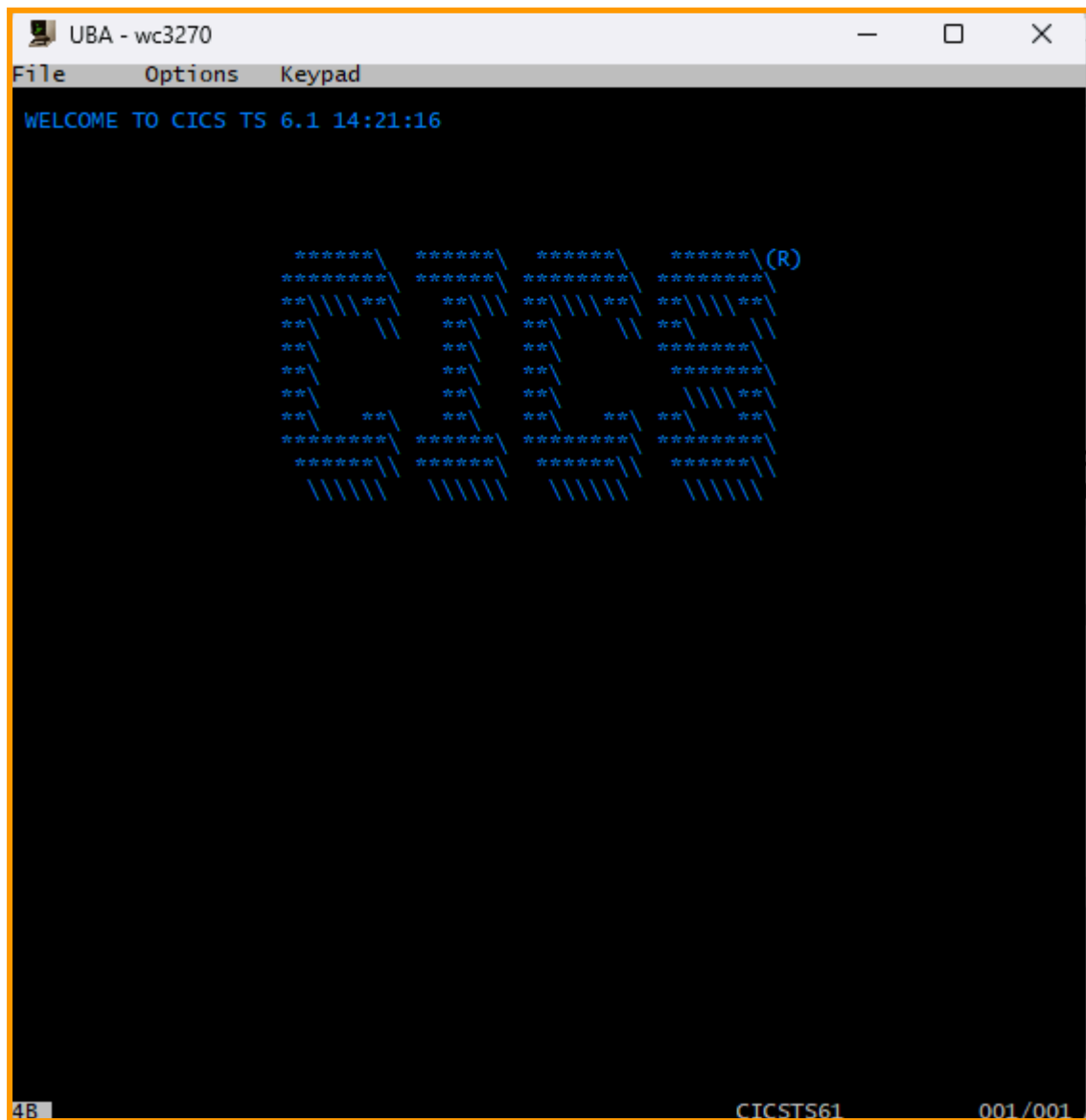
## CONFIGURACIÓN BÁSICA DE CICS

### – LAS TABLAS PROPIAS

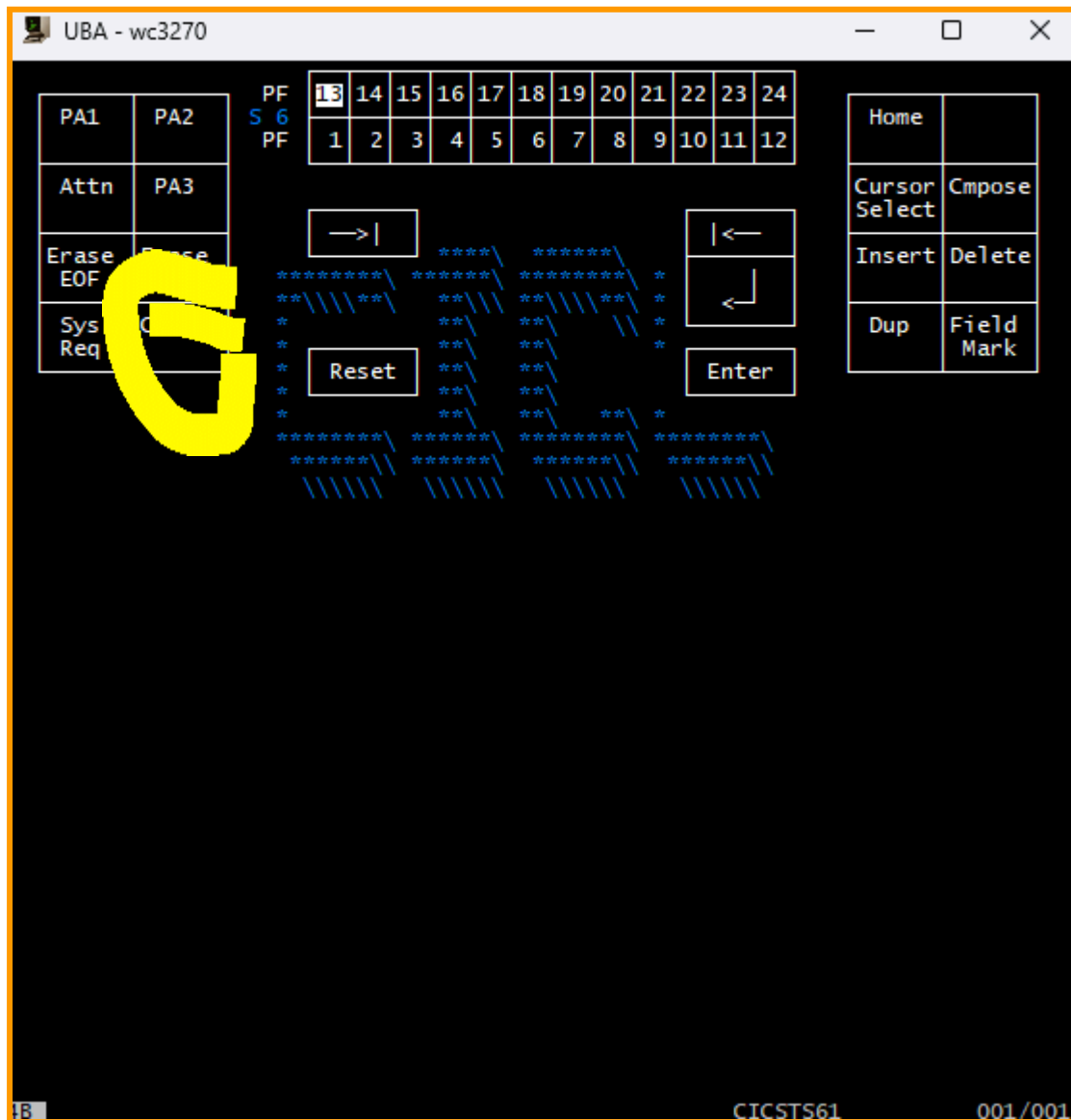
En CICS, las tablas propias son estructuras internas que definen cómo se comportan y se configuran los distintos componentes del sistema. Algunas de las más importantes incluyen:

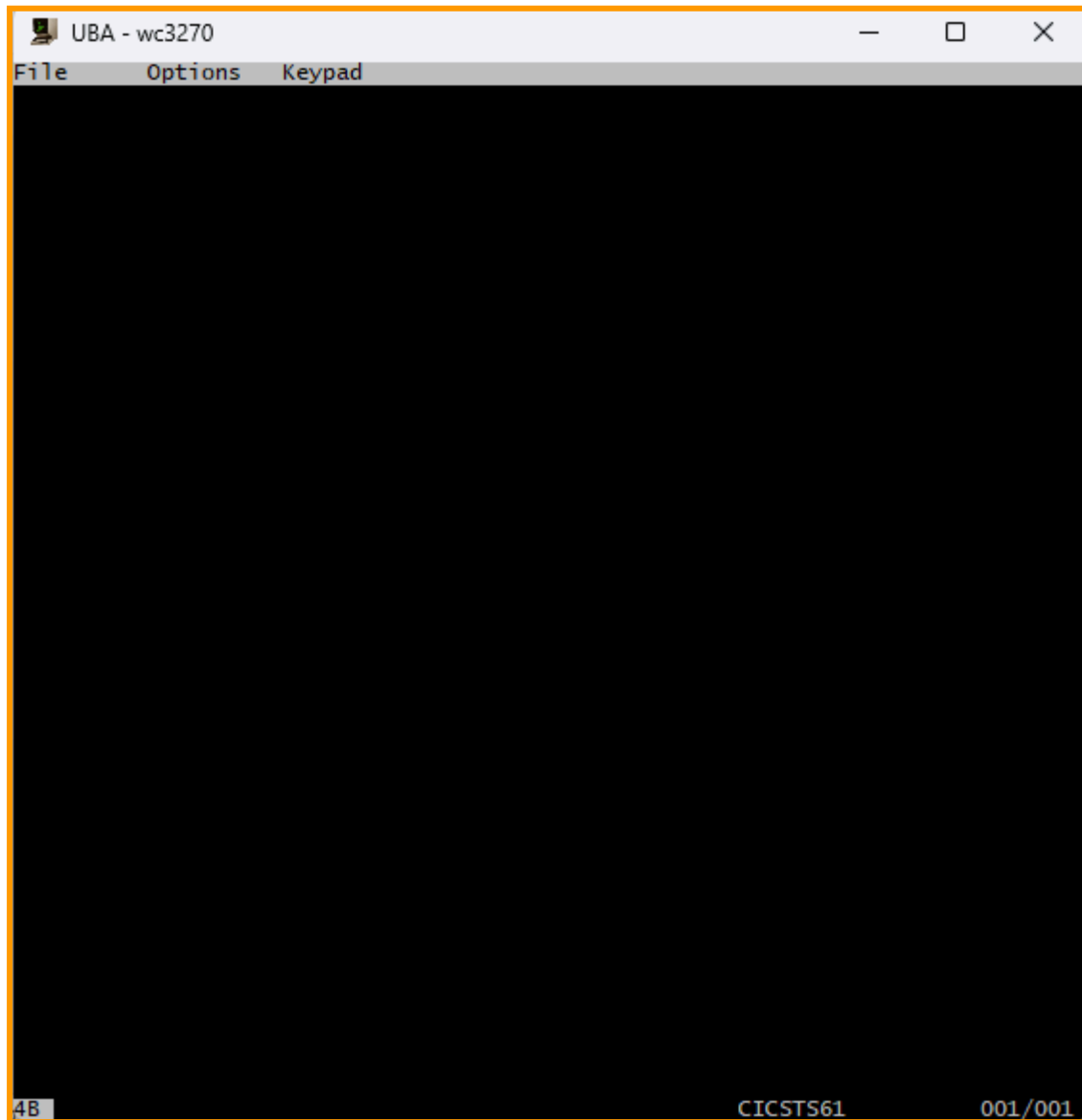
- **PCT (Program Control Table)**: define las transacciones disponibles y a qué programa están asociadas.
  - **PPT (Processing Program Table)**: describe los programas que pueden ejecutarse, sus nombres y características.
  - **FCT (File Control Table)**: especifica los archivos que CICS puede usar, como archivos VSAM.
  - **TCT (Terminal Control Table)**: configura los terminales o dispositivos conectados al sistema.
  - **DCT (Destination Control Table)**: se usa para definir destinos de impresión o colas de salida.
  - **TST (Temporary Storage Table)** y **TDQ (Transient Data Queue)**: gestionan almacenamiento temporal y colas de datos transitorios respectivamente.
-





LIMPIAR PANTALLA CON 'CLEAR' (dentro de opción KEYPAD)

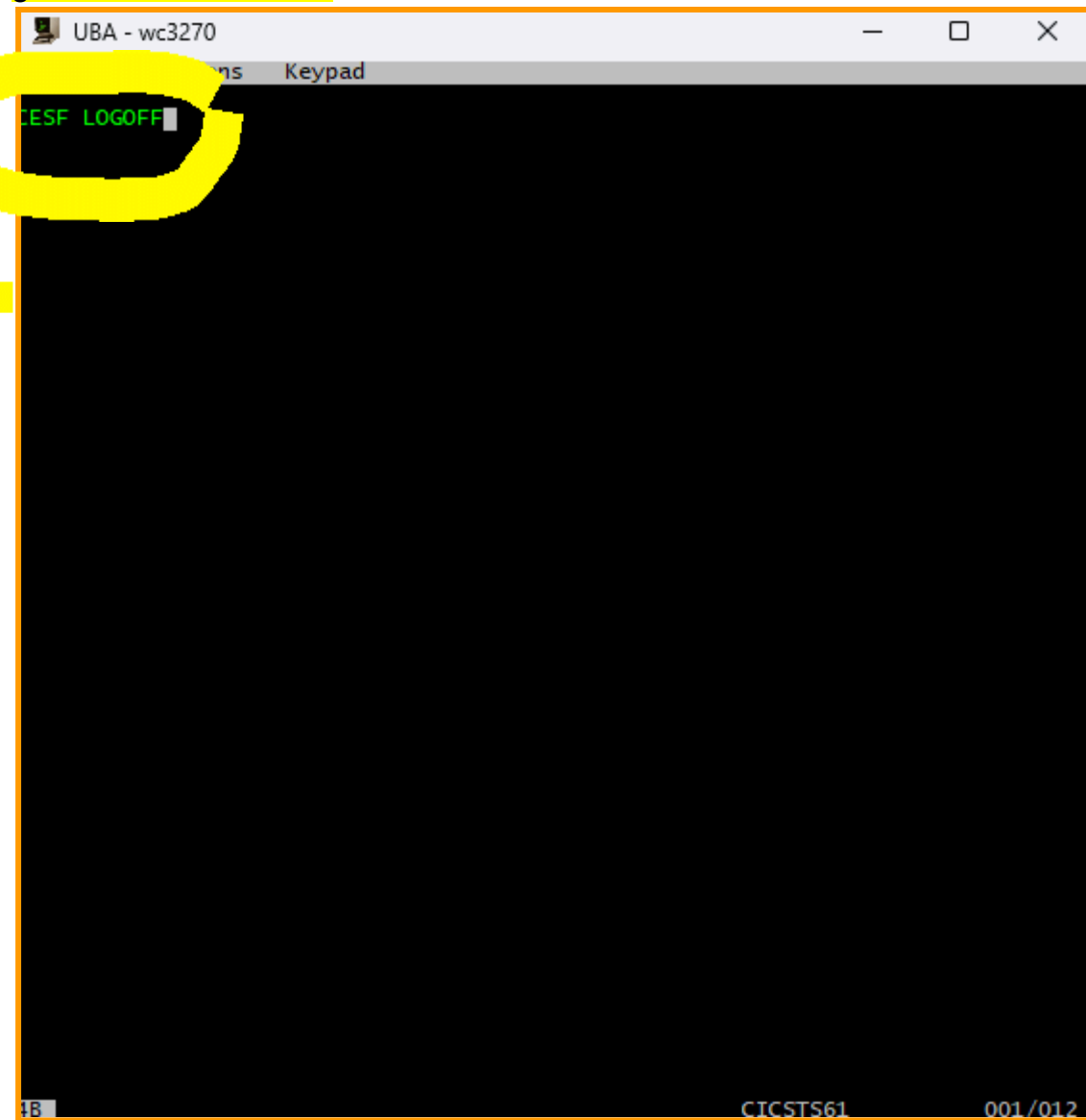




Recién en este espacio podrás ingresar la transacción que quieras arrancar.

---

## ¿ Cómo salir de CICS ? : CESF LOGOFF



**PRESIONAR 'ENTER' y sale de la Section**



## COMANDOS DE CICS

CICS ofrece una amplia gama de **comandos propios** que permiten a los programas interactuar con recursos del sistema, gestionar transacciones y controlar el flujo de ejecución. Estos comandos se escriben generalmente con la sintaxis EXEC CICS ... END-EXEC y se utilizan en lenguajes como COBOL, PL/I o C.

Aquí te dejo una lista de algunos comandos clave:

- **SEND / RECEIVE:** para mostrar datos en pantalla o recibir entradas del usuario.
- **READ / WRITE / DELETE:** para interactuar con archivos (por ejemplo, VSAM).
- **LINK / XCTL:** para transferir el control entre programas.
- **RETURN:** finaliza una transacción y devuelve el control al sistema.
- **ABEND:** termina una tarea de forma anormal, útil para manejo de errores.
- **ADDRESS / ASSIGN:** acceden a áreas de memoria o información del entorno.
- **INQUIRE / SET:** permiten consultar o modificar atributos de recursos como archivos, programas o terminales.
- 

Además, existen **transacciones administrativas (PROPIAS DE CICS)** como:

- **CECI:** (CICS EXECUTE COMMAND INTERPRETER) para probar comandos CICS directamente desde una terminal.
- **CEDA:** para definir recursos (programas, archivos, colas, etc.).
- **CEMT:** para gestionar recursos en tiempo de ejecución (por ejemplo, hacer un **newcopy** de un programa).

En general se deberá observar que, toda transacción que comienza con la letra '**C**' está reservada para uso exclusivo de CICS.

## ¿QUÉ ES EL NEWCOPY EN CICS?

El comando **NEWCOPY** en CICS se utiliza para **recargar una versión actualizada de un programa** en memoria sin necesidad de reiniciar la región de CICS. Es especialmente útil cuando se ha recompilado un programa y se quiere que CICS utilice la nueva versión inmediatamente.

Cuando se ejecuta un **CEMT SET PROGRAM**(nombre) **NEWCOPY**, CICS:

- Marca el programa como “no residente” en su tabla interna.
- La próxima vez que se invoque ese programa, lo cargará desde la biblioteca de carga (**loadlib**), trayendo así la versión más reciente.
- Si el programa está en uso (por ejemplo, en una sesión activa), la nueva copia se usará solo cuando finalicen esas sesiones.

Vamos con un ejemplo paso a paso de cómo hacer un **NEWCOPY** en CICS para actualizar un programa sin reiniciar la región:

### Supuestos

- Se tiene un programa llamado **PGMALCAB** que fue recompilado y cargado en la **loadlib** correspondiente.
- A efectos que CICS use esta nueva versión inmediatamente; se deberá seguir el siguiente:

### **Paso a paso**

#### **1. Verificar que el programa está definido en CICS**

mediante comando CEMT:

```
UBA - wc3270
file Options Keypad

I PROG(PGMALCAB)
STATUS: RESULTS - OVERTYPE TO MODIFY
Prog(PGMALCAB) Leng(0000000000) Cob Pro Ena Pri Ced
Resc(0000) Use(0000000000) Any Uex Ful Qua Cic

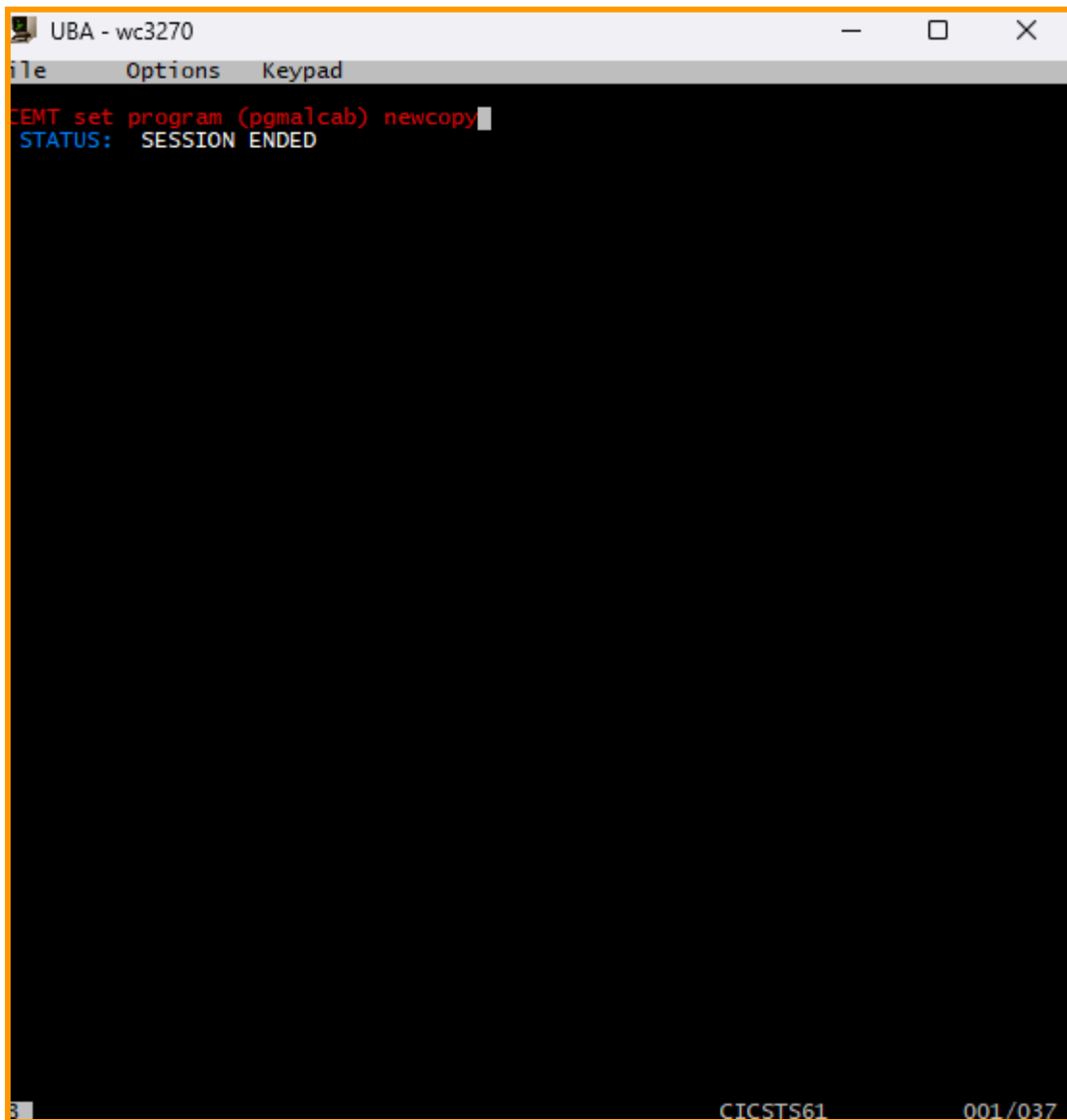
RESPONSE: NORMAL
PF 1 HELP 3 END 5 VAR 7 SBH 8 SFH 9 MSG 10 SB 11 SF

SYSID=S740 APPLID=CICST56
TIME: 14.16.15 DATE: 06/22/25

CICST561 001/020
```

Se escribirá NEW al final de la línea.

O ingresando directamente mediante comando SET:



```
UBA - wc3270
file Options Keypad
CEMT set program (pgmalcab) newcopy
STATUS:  SESSION ENDED
```

The screenshot shows a terminal window with a title bar 'UBA - wc3270' and standard window controls. Below the title bar is a menu bar with 'file', 'Options', and 'Keypad'. The main area of the terminal displays the command 'CEMT set program (pgmalcab) newcopy' in red text, followed by the status 'STATUS: SESSION ENDED' in blue text. At the bottom of the terminal, there is a status bar showing 'CICSTS61' and '001/037'.

Luego de presionar 'ENTER' en ambos casos, indicará a CICS que la próxima vez que se invoque **PGMALCAB**, lo cargue desde la biblioteca de carga (*LOADLIB*).

SIEMPRE verificar que el NEWCOPY fue exitoso; o sea que el largo en bytes del programa en tabla PPT se ha modificado (parámetro **Leng**).

## ¿CÓMO EJECUTAMOS LA COMPILACIÓN?

En entorno TSO:

**IMPORTANTE - PASO PREVIO:**      Alocar biblioteca **KC03XXX.CURSOS.COPYLIB**

con la misma estructura que **KC02788.ALU9999.COPYLIB**

a efectos que el **COPY COBOL** de cada **MAPA** compilado quede disponible en dicha biblioteca para que, posteriormente, lo pueda leer la compilación del programa que utilice dicho mapa.

---

# 1) Compilar el mapa asociado a la transacción:

- KC02788.ALU9999.FUENTE(COMPMAPA)  
KC03XXX.CURSOS.FUENTE

copiarlo

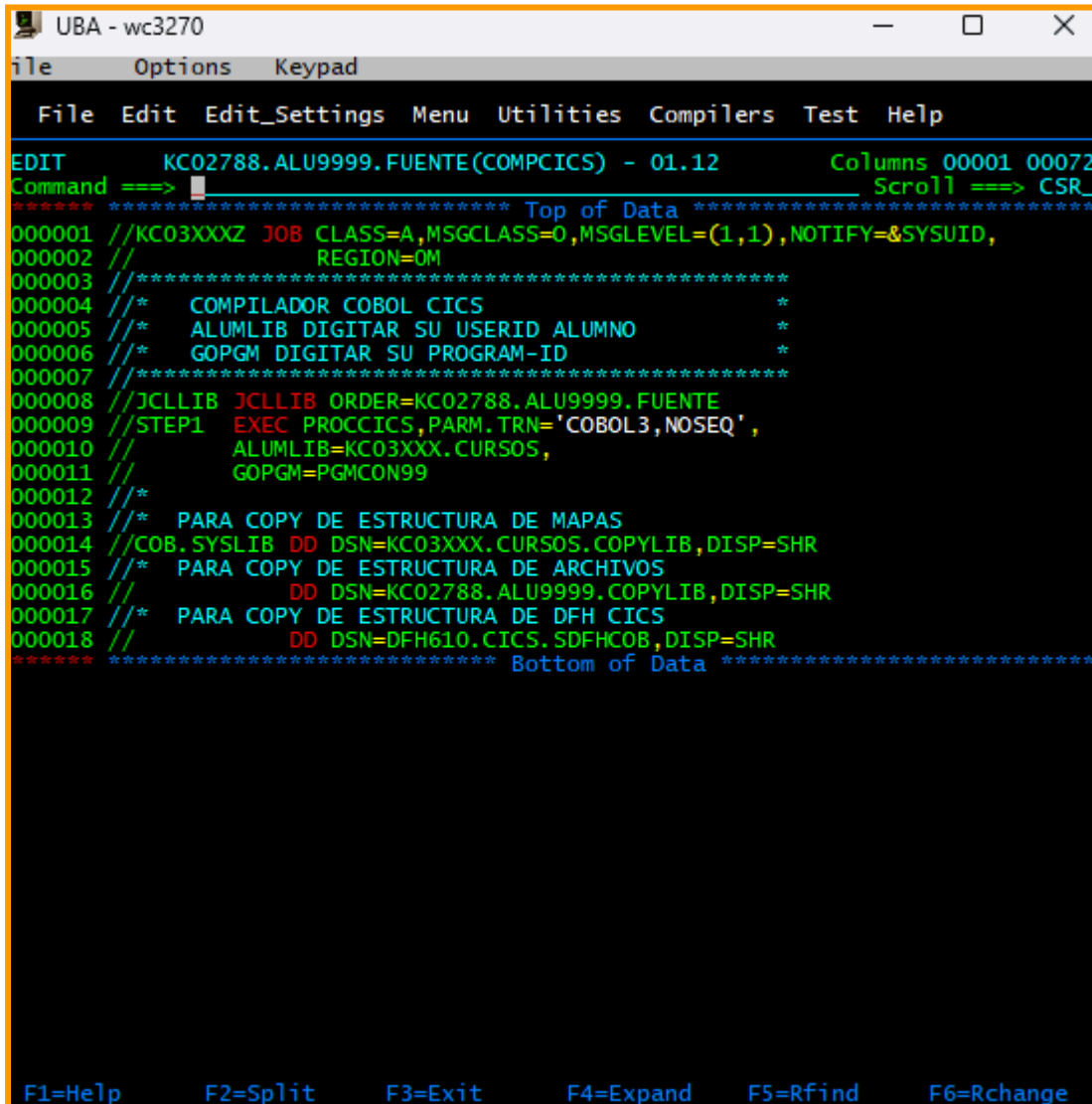
a

```
UBA - wc3270
file Options Keypad
File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT KC02788.ALU9999.FUENTE(COMPMAPA) - 01.02 Columns 00001 00072
Command ==> Scroll ==> CSR
***** Top of Data *****
000001 //KC03XXXA JOB CLASS=A,MSGCLASS=0,MSGLEVEL=(1,1),NOTIFY=&SYSUID
000002 //*****
000003 //* COMPILADOR MAPAS CICS *
000004 //* ALUMLIB DIGITAR SU USERID ALUMNO *
000005 //* MAPNAME DIGITAR SU NOMBRE DE MAPA *
000006 //*****
000007 //JCLLIB JCLLIB ORDER=KC02788.ALU9999.FUENTE
000008 //STEP1 EXEC PROCMAPA,
000009 // ALUMLIB=KC03XXX.CURSOS,
000010 // MAPNAME=MAP0099
***** Bottom of Data *****

F1=Help F2=Split F3=Exit F4=Expand F5=Rfind F6=Rchange
F7=Up F8=Down F9=Swap F10=Left F11=Right F12=Cancel
8 TS000008 004/015
```

## 2) Compilar el programa asociado a la transacción:

- KC02788.ALU9999.FUENTE(COMPCICS) copiarlo a KC03XXX.CURSOS.FUENTE



```

UBA - wc3270
file Options Keypad
File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT KC02788.ALU9999.FUENTE(COMPCICS) - 01.12 Columns 00001 00072
Command ==> Scroll ==> CSR
***** Top of Data *****
000001 //KC03XXXZ JOB CLASS=A,MSGCLASS=0,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000002 // REGION=OM
000003 //*****
000004 //* COMPILADOR COBOL CICS *
000005 //* ALUMLIB DIGITAR SU USERID ALUMNO *
000006 //* GOPGM DIGITAR SU PROGRAM-ID *
000007 //*****
000008 //JCLLIB JCLLIB ORDER=KC02788.ALU9999.FUENTE
000009 //STEP1 EXEC PROCCICS,PARM.TRN='COBOL3,NOSEQ',
000010 // ALUMLIB=KC03XXX.CURSOS,
000011 // GOPGM=PGMCON99
000012 //*
000013 //* PARA COPY DE ESTRUCTURA DE MAPAS
000014 //COB.SYSLIB DD DSN=KC03XXX.CURSOS.COPYLIB,DISP=SHR
000015 //* PARA COPY DE ESTRUCTURA DE ARCHIVOS
000016 // DD DSN=KC02788.ALU9999.COPYLIB,DISP=SHR
000017 //* PARA COPY DE ESTRUCTURA DE DFH CICS
000018 // DD DSN=DFH610.CICS.SDFHCOB,DISP=SHR
***** Bottom of Data *****
F1=Help F2=Split F3=Exit F4=Expand F5=Rfind F6=Rchange

```

## En entorno CICSTS61:

Antes de disparar la transacción asociada a la funcionalidad que se quiere testear;  
**RECORDATORIO:** NEW COPY del MAPA y del PROGRAMA

Luego se tipea la transacción + 'ENTER' y comienza la navegabilidad y el testeo.

---