COBOL

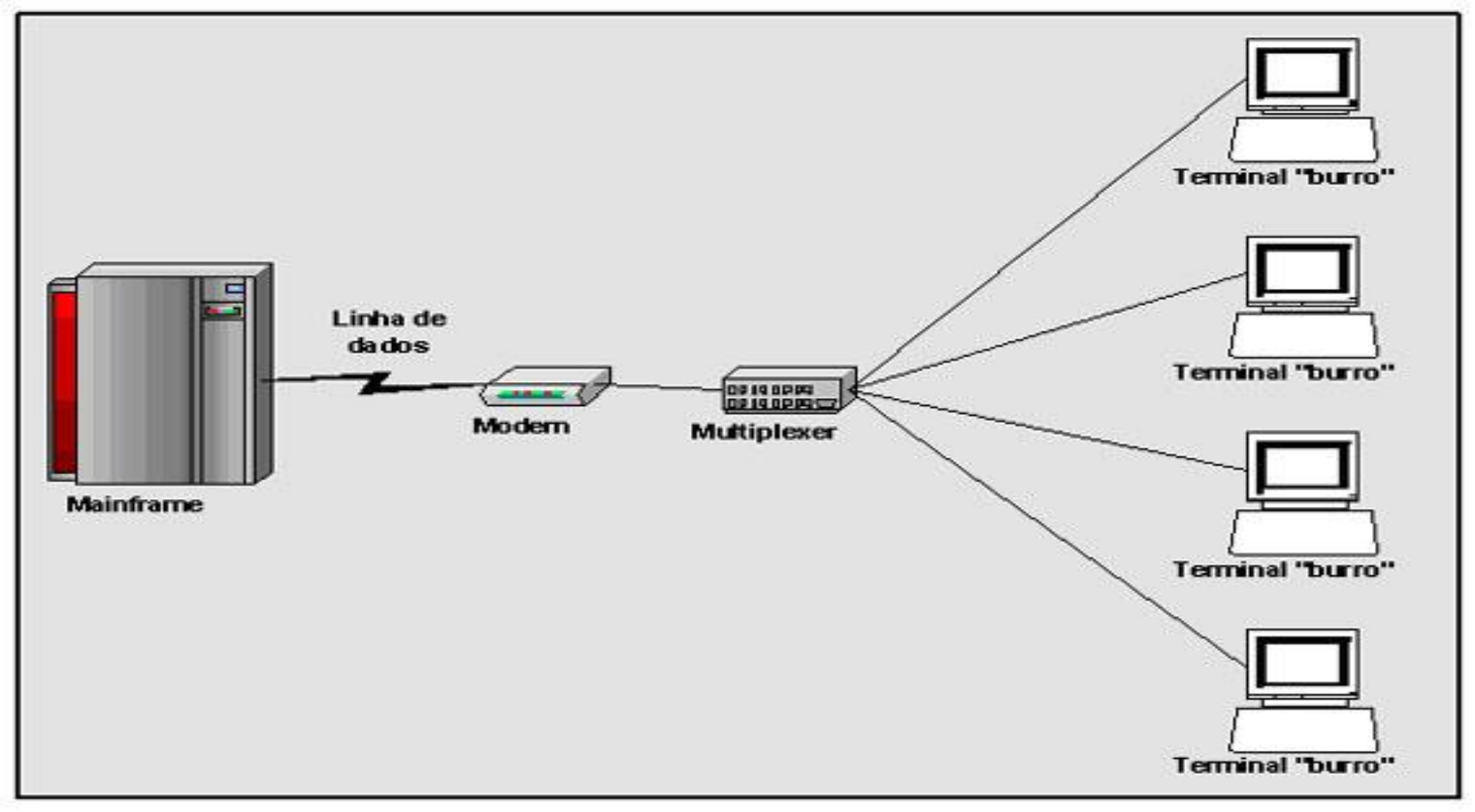
INTRODUÇÃO

ANDRE COSTA

Cobol

AQUI

Acesso Mainframe – Emuladores TN3270



Cobol

AQUI

Emulador PF teclado TN3270 p/ PC

PF1 ~ F1 - Ajuda PF2 ~ F2 - Liberado PF3 ~ F3 - Sai (Retorno tela anterior) PF4 ~ F4 - Liberado PF5 ~ F5 - Finaliza Sessão PF6 ~ F6 - Impressão PF7 ~ F7 - Paginação(pag. Anterior) PF8 ~ F8 - Paginação(próxima pag.) PF9 ~ F9 - Liberado PF10 ~ F10 – visualizar dados á direita

PF11 ~ F11 - visualizar dados á esquerda

PF12 ~ F12 - Liberado

Obs.:configurar teclado do emulador de acordo c/ambiente

Hopper

AQUI

Uma senhora nova-iorquina, Rear Admiral (Contra-Almirante) Grace Brewster Murray Hopper, sem dúvida merece ser mais conhecida entre programadores principalmente os "coboleiros".

Normalmente chamada simplesmente de Grace Hopper, ela foi a grande mentora intelectual na criação da linguagem COBOL.

Parece que nos preocupamos tanto com a memória das máquinas, que nos esquecemos de cuidar da nossa e passá-la para as novas gerações

Vamos lembrar um pouco da mulher águia, é ! Aquela que com seus óculos fundo de garrafa enxergava em 360 graus muitos anos na frente de seu tempo aquilo que os outros só viam quando ela mostrava.

A vovó que todos nós certamente teríamos orgulho em ter, também é considerada a mãe dos conceitos de biblioteca de rotinas e compilador, do costume de usar os termos bug e debug para a tarefa de identificar e eliminar erros de software e até mesmo do processamento de dados comercial moderno.

Definição

AQUI

A palavra COBOL é a abreviação de Commom Busines Oriented Language.

Esta é uma linguagem de computador orientada para negócios. As regras que comandam o uso da linguagem a fazem aplicável a problemas comerciais. Criada em 1959, tem passado por grandes e constantes aperfeiçoamentos, inclusive com versões
WINDOWS.

Todas as instruções são codificadas em inglês, em vez de códigos complexos. São programas mais extensos, porém mais claros e de mais rápidos entendimento e assimilação, não só da linguagem como dos programas escritos nela.

Segundo estimativas, a linguagem Cobol conta atualmente com aproximadamente 200 milhões de linhas de código, correspondendo por 70% das aplicações de negócios. Estima-se que há mais de 3 milhões de programadores Cobol empregados.

Nenhuma Ferramenta é excelente, todas são boas... Excelente são os profissionais que as usam....

Cobol morreu?

AQUI

Normalmente somos vistos como dinossauros. Nós, os profissionais Cobol, somos chamados de velhos e lentos, peças de museu. Quando se fala em Cobol logo somos associados a velhos computadores que ocupavam um prédio inteiro... e ainda ouvimos: Isso é coisa do passado o Cobol morreu..... contudo, temos:

Clube Cobol - Porque Cobol?

Pontos Fracos:
Ambiente de programação na sua maioria de difícil aprendizado
Uma fraqueza é que o Cobol é visto como estático, muito embora a linguagem seja muito dinâmica. Ele não é visto como elegante, no mundo orientado a objeto.
Não tem muita exposição. Visual Basic e Java são pesadamente promovidos e enormes quantidades de dinheiro são gastas em marketing. Há diversos vendedores de Cobol, porém nada aparece na imprensa, assim as pessoas concluem que a linguagem deve estar morrendo.

Pontos Fortes:
Confiabilidade, facilidade de manutenção, portabilidade, desempenho eficiente e disponibilidade

Divisões

AQUI

Todo programa COBOL consiste, obrigatoriamente, em 4 (quatro) divisões separadas.

IDENTIFICATION DIVISION

A IDENTIFICATION DIVISION serve para identificar o programa no computador e também proporciona informações documentais que são de suma importância para pessoas que não entendem nada de processamento e queiram analisar superficialmente o programa.

ENVIRONMENT DIVISION

A ENVIRONMENT DIVISION descreve o computador e os periféricos que serão utilizados pelo programa.

DATA DIVISION

A DATA DIVISION descreve os arquivos de entrada e saída que serão processados pelo programa, especificando seus formatos. Também define as áreas de trabalho e constantes necessárias para o processamento dos dados.

PROCEDURE DIVISION

É nesta divisão que o desenvolvedor descreverá o algoritmo do programa.
Esta divisão possui uma estrutura hierárquica e consiste de seções, parágrafos, sentenças e comandos.

Regras básicas

AQUI

REGRAS BÁSICAS

Os nomes de divisões e parágrafos devem ser codificados na margem A (coluna 8). Todas as outras declarações são codificadas a partir da margem B (coluna 12).
Cada Parágrafo termina com um ponto final.
Nós podemos representar a hierarquia do COBOL da seguinte forma:

Regras para formação de nomes:
De 1 até 30 caracteres
Não podem começar nem terminar com hífen
Conter pelo menos um caractere alfabético
Não podem ser palavra reservada do COBOL
Podem conter letras, números ou hífens mas nenhum caractere especial
Não são permitidos espaços em branco dentro de nomes de dados

Literais numéricas:
Máximo de 18 (dezoito) dígitos
Sinal ("+" ou "-") à esquerda do número
Ponto decimal, que não pode ser o último caractere

Identification Division

AQUI

A IDENTIFICATION DIVISION serve para identificar o programa no computador e também proporciona informações documentais que são de suma importância para pessoas que não entendem nada de processamento e queiram analisar superficialmente o programa.

IDENTIFICATION DIVISION ou ID DIVISION.
PROGRAM-ID. nome do programa.
[AUTHOR. nome do desenvolvedor.]

PROGRAM-ID (Program Identification)
Aqui deverá ser informado o nome do programa através do qual ele será identificado.
AUTHOR
Cláusula opcional onde pode ou não constar o nome do autor do programa

```
*-----------------------------------------------------------------*
*                   IDENTIFICATION DIVISION                       *
*-----------------------------------------------------------------*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. GMAPO094
 AUTHOR. DEBORAH DIAS SOUZA.
*                                                                 *
* SISTEMA       : SIGMA - SISTEMA DE MARKETING                    *
* OBJETIVO      : INCLUIR HISTORICO DO ANDAMENTO DA ACAO DE       *
*                 MARKETING - NIVEL COORDENACAO                   *
* ANALISTAS     : ROGERIO SILVA                                   *
* ESPECIFICADOR : ROGERIO SILVA                                   *
* PROGRAMADOR   : DEBORAH DIAS SOUZA                              *
* DATA          : 17/02/2006                                      *
* LINGUAGEM     : COBOL CICS/DB2                                  *
* VERSAO        : 001                                             *
```

Environment Division

AQUI

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.
OBJECT-COMPUTER.
SPECIAL-NAMES.
INPUT-OUTPUT SECTION.
FILE-CONTROL.

CONFIGURATION SECTION.

Esta seção destina-se a configuração do ambiente, ela é composta por três partes: SOURCE-COMPUTER, OBJECT-COMPUTER e SPECIAL-NAMES.
Uma identifica o computador onde foi confeccionado o programa e a segunda identifica o computador do ambiente de produção, ambas servem apenas para comentários haja visto que ambas deixaram de ser obrigatórias mas caso o programador opte em utiliza-las deve escrever sua sintaxe corretamente para não causar erros de compilação.
SPECIAL-NAMES especifica o sinal monetário, escolhe o tipo de ponto decimal, especifica caracteres simbólicos e possibilitar adaptar o programa para se comunicar com programas de outras linguagens.

Environment Division

AQUI

INPUT-OUTPUT SECTION.

Esta seção destina-se a configuração do ambiente de Leitura e Gravação, ela possui duas partes: FILE-CONTROL e I-O-CONTROL.
A primeira destina-se a especificação dos arquivos que o programa irá acessar. A segunda foi descontinuada nas versões mais atuais do compilador, valendo apenas para os ambientes de Mainframe

A cláusula SELECT terá a seguinte estrutura para arquivos seqüenciais:

```
SELECT nome-arquivo ASSIGN TO nome-externo
    ORGANIZATION IS SEQUENTIAL
    ACCESS MODE IS SEQUENTIAL
    FILE STATUS IS fs-arquivo.
```

Nome-arquivo indica o nome que será referenciado internamente pelo programa. Nome-externo indica onde os dados serão gravados, ou seja, o nome do arquivo, caso não seja especificado o caminho do arquivo será considerado o diretório corrente. Pode-se substituir este nome-externo por DISK e especificar o nome-externo na FD (será visto mais à frente) e se mesmo assim o nome-externo não for informado o compilador irá criar no diretório corrente de execução um arquivo com o mesmo nome-arquivo.

Environment Division

AQUI

ORGANIZATION IS SEQUENTIAL indica o tipo de organização do arquivo.
ACCESS MODE IS SEQUENTIAL o tipo de acesso (sendo que este é o único tipo de acesso permitido para esta organização).
FILE STATUS IS fs-arquivo representa uma referência de um campo da Working-Storage section que será atualizada a cada operação feita no arquivo, indicando qual é a situação atual do arquivo, é desta forma que o programa pode fazer testes para saber se pode ou não continuar a execução sem problemas.

```
*-----------------------------------------*
 CONFIGURATION                    SECTION.
*-----------------------------------------*
 SPECIAL-NAMES.
     DECIMAL-POINT IS COMMA.
*-----------------------------------------*
 INPUT-OUTPUT                     SECTION.
*-----------------------------------------*
 FILE-CONTROL.
     SELECT DLDFD001              ASSIGN TO UT-S-DLDFD001
            FILE STATUS IS DLDFD001-FS.
*
```

Data Division

AQUI

Divisão voltada única e exclusivamente à definição de estruturas de registros, variáveis e constantes do programa, ou seja, uma área de alocação de memória para todo o espaço necessário ao seu programa.
As duas principais seções são: a FILE SECTION e a WORKING-STORAGE SECTION

FILE SECTION

Seção que define a estrutura dos arquivos de dados. Esta definição envolve a descrição do arquivo e seus respectivos registros. Para cada SELECT definido temos uma definição de arquivo na FILE SECTION.

WORKING-STORAGE SECTION

Seção que descreve e armazena numa área de memória todos os dados, informações,variáveis e constantes, com valores definidos ou não, a serem manipulados pelo programa. É composta de itens de grupo e itens elementares.

Data Division

AQUI

```
FILE SECTION.
      FD nome-arquivo
            [ RECORD CONTAINS nn  CHARACTERS ]

            [ LABEL RECORD  IS  { OMITTED, STANDARD } ]

            [ VALUE OF  FILE-ID valor-identificação-arquivo ]  .

      01 nome-de-registro-arquivo .
            [ nro-nivel
                  [ nome-campo ou FILLER ]
                  [ REDEFINES  nome-de-dado  ]
                  [ PIC   tipo(tamanho) ]
                  [ OCCURS  nro-inteiro TIMES ]  ] .
```

Data Division

AQUI

RECORD CONTAINS => especifica o tamanho do registro de dados. O tamanho d registro é determinado pelo soma do número de caracteres de todos os itens elementares subordinados ao registro.

LABEL RECORD=> especifica se existe rótulo presente no arquivo. Omitted especifica que não existe rótulo explícito(arquivos de impressão). Standard especifica que existem rótulos e estão conforme as especificações do sistema operacional(disco).

VALUE OF FILE-ID=> identifica o nome do arquivo no meio externo.

NRO-NIVEL=> são números entre 01 e 49 que permitem a estruturação de um registro lógico, pela subdivisão deste registro. Uma vez que uma subdivisão tenha sido especificada(item de grupo), ela pode ser ainda mais subdividida(itens elementares), para permitir uma referencia mais detalhada. Item elementar é a subdivisão fundamental de um registro, que não é mais subdividido. Um registro pode ser constituído de uma seqüência de itens elementares ou pode ser somente um item elementar.

Um item de grupo é uma seqüência de um ou mais itens elementares ou também de um ou mais itens de grupo.Uma descrição de um registro sempre começa pelo número de nível 01.

NOME-CAMPO=> nome definido pelo programador que não pode ser repetido dentro do fonte do programa, pode ter até 30(trinta) caracteres e não pode ser igual a alguma palavra reservada da sintaxe do COBOL.

FILLER=> palavra reservada do COBOL que serve para reservar uma determinada quantidade de bytes em um arquivo ou na memória.

REDEFINES=> cláusula utilizada para redefinir um item de grupo e/ou item elementar em partes menores ou em uma imagem diferente.

Exemplo File Section

```
EXEMPLO: 01
FD  BCI022E
    RECORDING IS F
    RECORD 17.

01  BCI022E-REGISTRO.
    10 CD-EST-CTR              PIC S9(4) USAGE COMP.
    10 CD-DEPE-EXEC            PIC S9(4) USAGE COMP.
    10 CD-REF-BASE-CMB         PIC S9(9) USAGE COMP.
    10 CD-SEQL-REG             PIC S9(17)V USAGE COMP-3.

EXEMPLO: 02

FD  DLDFD001
    LABEL  RECORD  STANDARD
    RECORDING MODE IS V
    RECORD  VARYING IN SIZE
    FROM 1 TO 32750 CHARACTERS
    DEPENDING ON WSS-TAM-F001S.
01  DLDFD001-REG-FD        PIC  X(32750).
```

Working Storage Section

AQUI

Utilizada para descrever registros e campos auxiliares de trabalho, tais como: totalizadores, contadores, flags etc. Aqui pode ser definido também a formatação de relatórios, cabeçalhos e linhas detalhes.

NIVEL 77 : VARIAVEIS ELEMENTARES (NAO SAO ESTRUTURAS)
NIVEL 88 : FLAGS QUE SE TORNAM VERDADEIRAS QUANDO A VARIAVEL
DO NIVEL IMEDIATAMENTE ANTERIOR ASSUME O VALOR
ASSOCIADO ‘A FLAG, E FALSAS CASO CONTRARIO
OUTROS NIVEIS : ESTRUTURAS DE DADOS

PICTURE X : VARIAVEIS ALFANUMERICAS
PICTURE 9 : VARIAVEIS NUMERICAS
PICTURE Z : VARIAVEIS DE DISPLAY (O DIGITO APENAS APARECE SE
‘A ESQUERDA HOUVEREM DIGITOS DIFERENTES DE ZERO)
PICTURE S9, -9, -Z, 9-, Z- : VARIAVEIS COM SINAL
PICTURE 9(i)V9(d) : VARIAVEIS COM PARTE INTEIRA
(i DIGITOS) E PARTE DECIMAL (d DIGITOS)
PICTURE 9 COMP : VARIAVEIS COMPACTADAS
(MAIS DE UM DIGITO POR BYTE)

Exemplos de Working

AQUI

```
WORKING-STORAGE SECTION.

77  WS-ARRED-ESC          PIC 9(13)V.
77  W-MSG                 PIC X(70)      VALUE SPACES.
77  WS-IMP-01             PIC 9(15)      VALUE ZEROS.
77  WS-VFINANC            PIC 9(14)V99.
77  W-QNEGOC              PIC S9(09) COMP VALUE ZEROS.
77  W-VPUORDEM-MIN        PIC S9(10)V99 COMP-3.
77  W-CONT                PIC S9(04)    COMP.
77  STATUS-FICH           PIC X(02).
    88  STATUS-FIM                      VALUE '10'.
    88  STATUS-OK                       VALUE '00'.
    88  STATUS-DUP                      VALUE '22'.
    88  STATUS-INEXISTENTE              VALUE '23'.
```

Exemplos de Working

AQUI

```
WORKING-STORAGE SECTION.

01 LINHAS-TEXTO.
   05 LINHA-TEXTO OCCURS 13 TIMES.
      10 NUM-LINHA          PIC  9(2).
      10 TEXTO-LINHA        PIC X(78).

01  WS-DATA-DIA             PIC 9(06).
01  FILLER REDEFINES WS-DATA-DIA.
    10  WS-ANO               PIC 9(02).
    10  WS-MES               PIC 9(02).
    10  WS-DIA               PIC 9(02).
```

LINKAGE SECTION

AQUI

É a divisão do cobol que especifica as ações necessárias para o processamento de dados em geral: controle de execução, entrada e saída, movimento dos dados, etc. Sua função é descrever tais procedimentos necessários para resolução de um problema dado.

Os procedimentos descritos na PROCEDURE DIVISION são escritos em "STATEMENTS" e estão, geralmente, agrupados em parágrafos cujos nomes são fornecidos pelo programador. Tais parágrafos podem, ainda, estarem agrupados em SECTIONS também criadas pelo programador.

```
PROCEDURE DIVISION.
000-INICIAIS.
       PERFORM 000-ABRE-ARQUIVO
       PERFORM 100-LER-ARQUIVO
       PERFORM 400-FIM
000-ABRE-ARQUIVO.
     OPEN INPUT DADOS
     OUTPUT LISTAG.
100-LER-ARQUIVO.
      READ ARQALUN AT END
            MOVE "FIM" TO WS-CONTROLE
      ADD 1 TO WS-CONT-LIDOS
      MOVE REG-ALUN TO WS-REG-ALUN
```

Procedure Division

AQUI

É a divisão do cobol que especifica as ações necessárias para o processamento de dados em geral: controle de execução, entrada e saída, movimento dos dados, etc. Sua função é descrever tais procedimentos necessários para resolução de um problema dado.

Os procedimentos descritos na PROCEDURE DIVISION são escritos em "STATEMENTS" e estão, geralmente, agrupados em parágrafos cujos nomes são fornecidos pelo programador. Tais parágrafos podem, ainda, estarem agrupados em SECTIONS também criadas pelo programador.

```
PROCEDURE DIVISION.
000-INICIAIS.
      PERFORM 000-ABRE-ARQUIVO
      PERFORM 100-LER-ARQUIVO
      PERFORM 400-FIM
000-ABRE-ARQUIVO.
    OPEN INPUT DADOS
    OUTPUT LISTAG.
100-LER-ARQUIVO.
     READ ARQALUN AT END
          MOVE "FIM" TO WS-CONTROLE
     ADD 1 TO WS-CONT-LIDOS
     MOVE REG-ALUN TO WS-REG-ALUN
```

Procedure Division

Vários Comandos para o processamento?

IF E ELSE END-IF	– INSTRUÇÕES PARA DEFINIR CONDIÇÕES DENTRO DO PROGRAMA
PERFORM	– PARA CHAMAR UMA PROCEDURE
PERFORM UNTIL	– PARA EXECUTAR UM LOOP DE INSTRUÇÕES
MOVE	– USADO PARA ATRIBUIR VALORES A VARIÁVEIS
ADD	– USADO PARA ATRIBUIR VALORES A VARIÁVEIS
COMPUTE	– USADO PARA EFETUAR CALCULOS DIVERSOS
EVALUATE	– USADO PARA TOMAR DECISÕES AGRUPADAS
ACCEPT	– USADO PARA CAPTURAR UM DADO DIGITADO OU NO SISTEMA
STOP RUN	– FINALIZA O PROGRAMA
DISPLAY	– MOSTRA DADOS NA TELA
CALL	– CHAMA UM PROGRAMA
GOBACK	– VOLTA PARA O PROGRAMA CHAMADOR
INPUT	– ABRE UM ARQUIVO PARA LEITURA
OUTPUT	– CRIA UM ARQUIVO
I-O	– ABRE UM ARQUIVO PARA GRAVACAO
READ	– LE UM ARQUIVO
WRITE	– GRAVA UM ARQUIVO OU LINHAS
REWRITE	– REGRAVA UM ARQUIVO OU LINHAS
DELETE	– DELETA UM ARQUIVO OU LINHAS
CLOSE	– FECHA UM ARQUIVO

Procedure Division

Vamos praticar?

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO-WORLD.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 PROCEDURE DIVISION.
000-INICIO.
        DISPLAY “HELLO WORD".
•STOP RUN.
```

Cobol CICS

AQUI

CICS: È um dos mais conhecidos gerenciador de Transações para o mainframe, com ele podemos gerenciar todos os componentes que estão relacionados a uma transação.

Existem duas formas básicas de executar um programa Online: via LINK e via START. Para iniciar um programa ONLINE, é necessário que ele tenha uma transação associada e que os diversos objetos a ele associados estejam recenseados no CICS, nomeadamente:

NOME DO PROGRAMA: FSWPO001
NOME DA TRANSAÇÃO: FS01
NOME DO PLANO DB2: FSWPLAN

Para as aplicações COBOL/CICS com Java, em JABOL (JAVAXCOBOL), utilizamos somente alguns comandos CICS que vamos ver abaixo:

```
DFHCOMMAREA – RESPONSÁVEL PELA COMUNICAÇÃO DE DADOS ONLINE
LINK        - RESPONSÁVEL PELA CHAMADA DE PROGRAMAS ONLINE
RETURN      - RESPONSÁVEL PELA FINALIZAÇÃO DO PROGRAMA ONLINE
SYNCPOINT   - RESPONSÁVEL PELA EFETIVAÇÃO DA TRANSAÇÃO
SYNCPOINT ROLLBACK – RESPONSÁVEL POR DESFAZER A TRANSAÇÃO
```