









IBM System 360 on Modern Hardware and Windows OS

Objetivo:

Simular um **IBM System/360** e correr um pequeno programa em **Fortran IV**, como se estivéssemos em 1965, , usando o emulador **Hercules** no **Windows 11**. 

1. O que vais precisar:

1.  Windows 11 (já tens)
2.  Hercules Emulator (grátis)
3.  Um sistema operacional antigo (ex: MVS ou DOS/360)
4.  Compilador Fortran (vem incluído no sistema)
5.  Um programa exemplo em Fortran


🧩 2. Instalar o Emulador Hercules

Passos:

1. Vai ao site oficial: <https://github.com/SDL-Hercules-390/hyperion/releases>
 2. Descarrega a última versão do .zip (ex: Hyperion-Release-Win64.zip)
 3. Extrai para uma pasta simples como C:\Hercules
 4. Abre o ficheiro hercules.exe para testar: deve aparecer uma consola com mensagens técnicas.
-

📁 3. Usar um Sistema Operativo antigo (pré-configurado)

Para facilitar a experiência, usa uma imagem já pronta:

- Recomendo o **TK4-** (versão simplificada e gratuita do MVS Turnkey):
 Download direto: <http://wotho.ethz.ch/tk4-/tk4- v1.00 current.zip>

(importante: este link parece não estar a funcionar, a 19abr2025: foi preciso pesquisar pelo "TK4-" no google, e descarregar de outro local).

Instalação:

1. Extrai o .zip para C:\TK4-
 2. Vai até à pasta C:\TK4-\windows e corre mvs.bat
 - Isto vai arrancar o Hercules com a imagem MVS
 - Também abre um terminal de operador (Hercules console)
-

💻 4. Aceder ao MVS

O sistema arranca como um mainframe real. Para interagires:

1. Abre o **3270 Terminal Emulator** (emulador de terminal IBM)
 - Recomendo o **WC3270** (gratuito):
<https://github.com/watsonbox/wc3270/releases>
2. Liga ao sistema com:
3. Hostname: 127.0.0.1df
4. Port: 3270

● Deverás ver um login tipo "TSO/E READY"

5. Escrever um programa em Fortran (como nos anos 60!)

Exemplo: "HELLO, WORLD"

1. Cria um ficheiro .jcl com este conteúdo (chamado hello.jcl, por exemplo):

```
//HELLOJOB JOB (123),'FORTRAN TEST',CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
```

```
//STEP1 EXEC FORTG
```

```
//FORT.SYSIN DD *
```

```
WRITE(6,100)
```

```
100 FORMAT(' HELLO, FORTRAN ON THE MAINFRAME!')
```

```
STOP
```

```
END
```

```
/*
```

```
//
```

2. Usa um editor como o Notepad++ para salvar o ficheiro com codificação ASCII.
3. Copia o ficheiro para a pasta C:\TK4-\public (partilhada com o MVS).

6. Submeter o job no mainframe

1. No terminal 3270:
2. tso submit 'public.hello.jcl'
3. Espera uns segundos, depois escreve:
4. tso sdsf
5. Acede ao output do teu job:
 - Selecciona o teu job com S
 - Vê o output em JES2: deverás encontrar a mensagem HELLO, FORTRAN ON THE MAINFRAME!

Curiosidades:

- O compilador Fortran em uso é de 1971!
- O sistema operativo MVS era usado por bancos e governos nos anos 70 e 80.
- O JCL (Job Control Language) é a forma "antiga" de correr programas.

Dica pedagógica






Pensa no mainframe como um enorme "batch processor", onde programadores submeteram cartões perfurados que eram processados em fila. A tua hello.jcl é uma versão digital desses cartões!



Objetivo:

Simular um **IBM System/360** e correr um pequeno programa em **Fortran IV**, como se estivéssemos em 1965! (no Windows):

1. O que vais precisar:

1.  Windows 11 (já tens)
 2.  Hercules Emulator (grátis)
 3.  Um sistema operacional antigo (ex: MVS ou DOS/360)
 4.  Compilador Fortran (vem incluído no sistema)
 5.  Um programa exemplo em Fortran
-

2. Instalar o Emulador Hercules

Passos:

1. Vai ao site oficial: <https://github.com/SDL-Hercules-390/hyperion/releases>
 2. Descarrega a última versão do .zip (ex: Hyperion-Release-Win64.zip)
 3. Extrai para uma pasta simples como C:\Hercules
 4. Abre o ficheiro hercules.exe para testar: deve aparecer uma consola com mensagens técnicas.
-

3. Usar um Sistema Operativo antigo (pré-configurado)

Para facilitar a experiência, usa uma imagem já pronta:

- Recomendo o **TK4-** (versão simplificada e gratuita do MVS Turnkey):
 Download direto: http://wotho.ethz.ch/tk4-/tk4-_v1.00_current.zip

Instalação:

1. Extraí o .zip para C:\TK4-
 2. Vai até à pasta C:\TK4-\windows e corre mvs.bat
 - Isto vai arrancar o Hercules com a imagem MVS
 - Também abre um terminal de operador (Hercules console)
-

4. Aceder ao MVS

O sistema arranca como um mainframe real. Para interagires:

1. Abre o **3270 Terminal Emulator** (emulador de terminal IBM)
 - Recomendo o **WC3270** (gratuito):
<https://github.com/watsonbox/wc3270/releases>
2. Liga ao sistema com:
3. Hostname: 127.0.0.1
4. Port: 3270

● Deverás ver um login tipo "TSO/E READY"

5. Escrever um programa em Fortran (como nos anos 60!)

Exemplo: "HELLO, WORLD"

1. Cria um ficheiro .jcl com este conteúdo (chamado hello.jcl, por exemplo):

```
//HELLOJOB JOB (123),'FORTRAN TEST',CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
```

```
//STEP1 EXEC FORTG
```

```
//FORT.SYSIN DD *
```

```
WRITE(6,100)
```

```
100 FORMAT(' HELLO, FORTRAN ON THE MAINFRAME!')
```

```
STOP
```

```
END
```

```
/*
```

```
//
```

2. Usa um editor como o Notepad++ para salvar o ficheiro com codificação ASCII.
 3. Copia o ficheiro para a pasta C:\TK4-\public (partilhada com o MVS).
-



6. Submeter o job no mainframe

1. No terminal 3270:
2. tso submit 'public.hello.jcl'
3. Espera uns segundos, depois escreve:
4. tso sdsf
5. Acede ao output do teu job:
 - Selecciona o teu job com S
 - Vê o output em JES2: deverás encontrar a mensagem HELLO, FORTRAN ON THE MAINFRAME!



Curiosidades:

- O compilador Fortran em uso é de 1971!
- O sistema operativo MVS era usado por bancos e governos nos anos 70 e 80.
- O JCL (Job Control Language) é a forma "antiga" de correr programas.



Dica pedagógica

Pensa no mainframe como um enorme "batch processor", onde programadores submeteram cartões perfurados que eram processados em fila. A tua hello.jcl é uma versão digital desses cartões!



Vamos agora recriar um verdadeiro clássico académico dos anos 60 no **IBM System/360**: um programa **Fortran IV** que gera números aleatórios e calcula o **desvio padrão**. 🧮 ✨

⚠️ Atenção: o Fortran antigo (anos 60/70) não tem uma função moderna como `RANDOM()`, mas podemos simular números pseudoaleatórios com truques matemáticos simples — por exemplo, usando o método **Linear Congruential Generator (LCG)**.

📌 O que o programa vai fazer:

1. Lê **N** (quantidade de números a gerar)
 2. Gera N números pseudoaleatórios entre 0 e 1
 3. Calcula a média dos valores
 4. Calcula o desvio padrão
 5. Imprime os resultados
-

Ficheiro JCL: desvio.jcl

Guarda este conteúdo como desvio.jcl (ASCII puro), e coloca em C:\TK4-\public:

```
//DESVIO JOB (123), 'FORTRAN STDEV', CLASS=A, MSGCLASS=A, MSGLEVEL=(1,1)
//STEP1 EXEC FORTG
//FORT.SYSIN DD *
    INTEGER N, I
    REAL X(100), SUM, MEAN, SD, SEED
    READ(5,100) N
    100 FORMAT(I3)

C      Inicializa
      SEED = 0.5
      SUM = 0.0

C      Geração dos números e soma
      DO 10 I = 1, N
          SEED = MOD(16807.0 * SEED, 1.0)
          X(I) = SEED
          SUM = SUM + X(I)
      10 CONTINUE

      MEAN = SUM / N

C      Calcula desvio padrão
      SD = 0.0
      DO 20 I = 1, N
          SD = SD + (X(I) - MEAN)**2
      20 CONTINUE

      SD = SQRT(SD / N)

      WRITE(6,200) N, MEAN, SD
      200 FORMAT('N = ', I3, ' MEDIA = ', F6.4, ' DESVIO PADRAO = ', F6.4)
      STOP
      END

/*
//FORT.SYSIN DD *
    10
/*
```

Como funciona o gerador aleatório?

$SEED = MOD(16807.0 * SEED, 1.0)$

- Esta fórmula usa o multiplicador 16807 (conhecido do método LCG)
- O $MOD(..., 1.0)$ mantém o valor entre 0 e 1 (parte fracionária)
- É um truque clássico para simular aleatoriedade nos mainframes sem instruções modernas

Como correr

1. Coloca o ficheiro em C:\TK4-\public
2. Liga ao terminal 3270
3. Executa:
4. `tso submit 'public.desvio.jcl'`
5. Vai ao SDSF:
6. `tso sdsf`
7. Abre o output do job e verás algo como:
8. `N = 10 MEDIA = 0.4873 DESVIO PADRAO = 0.2779`

Queres alterar o intervalo dos números (ex: entre 5 e 10)?

Basta modificar esta linha:

$X(I) = 5.0 + SEED * 5.0$

Assim, os números aleatórios passam a estar entre **5.0 e 10.0!**