Outline

IBM

- What is the Year 2000 problem?
- What does ″Year 2000 ready″ mean?
- Year 2000-ready products
- How to solve the Year 2000 problem
- Y2K conversion and language migration
- How to make your applications Y2K ready
- DFSORT and the century window
- COBOL program samples
- Testing your applications
- Look out!
- How to get more information

What is the Year 2000 problem?

IBM

- Date information stored with only 2 digits
- Incorrect century
  - 04 assumed to be 1904
- Arithmetic calculation
  - A person born in 1950 calculated to be -50 years old
  - Due date of 30 year mortgage started in 1990 could be calculated to be 1920, paid off!
- Sequence
  - 12/31/99 comes after 01/01/00
  - IMS database updates will fail without V5 and fix
- Leap Year
  - If divisible by 100, then not a leap year, right?
  - Except 2000 is a leap year

What is the Year 2000 problem?

IBM

For non-application code:

- **Archiving and deleting of old datasets**
  - **They might not be old on Jan 1 2000!**
- **Conventions of '99365' meaning keep forever**
  - **That date is only 3 years and 40 days away**
- **Expiration dates on tape labels**
- **Homemade dataset managers**
- **Any and all system software that has not solved the problem**

What does "Year 2000 ready" mean?

IBM

Year 2000 ready Product

The Product, when used in accordance with its associated documentation, is capable of correctly processing, providing and/or receiving date data within and between the twentieth and twenty-first centuries, provided that all products (for example, hardware, software and firmware) used with the Product properly exchange accurate date data with it.

- **This can be applied to your applications as well**

- **More simply, ask yourself:**

  - **Will your applications run after 1999?**

  - **Will they produce the correct output after 1999?**

Year 2000-ready products

IBM

- See the supplied book, GC28-1251
- Make sure that your hardware is ready
- Make sure that your system software is ready
- Examples:
  - CICS V3.3 or earlier will not run across midnight 12/31/1999
    - Might be able to do a complete ice-cold rebuild/restart for V3.3
  - IMS V4 and earlier will fail the first database update of 2000
    - 2-digit year dates in IMS Log records
    - IMS V5 with maintenance uses century window
  - OS/VS COBOL and VS COBOL II are NOT Year 2000 ready

Year 2000-ready products

IBM

- Languages are a whole different animal
  - Non-ready System Software will fail regardless of your applications
- We know of no problems with IBM languages running after 1999
  - For COBOL, only COBOL for MVS & VM and Language Environment are ready
- They will run, but old languages only give you a 2-digit year date!
  - NO 4-digit year date COBOL language in:
    - OS/VS COBOL or
    - VS COBOL II
- Your applications need century window and 4-digit year date functions
  - With only 2-digit year dates and no window, then fail!
- Your applications need to be changed
  - 90% of financial applications need to be changed
  - 75% of other applications need to be changed

How to solve the Year 2000 problem

IBM

1. **Replace non-ready hardware**

2. **Replace non-ready system software**

3. **Fix all of your applications and data**

4. **Link and run applications with**

Language Environment for MVS & VM

5. **Test**

That's it!

Y2K conversion and language migration

IBM

- **Share the same process, and most of the same steps**
  - **Planning**
  - **Inventory**
  - **Ownership**
  - **Status**
  - **(Source conversion/Year2000 changes)**
  - **Compile, link**
  - **Unit Test**
  - **System Test**

Combine and save money and time!

- **"Collateral Benefit"**

How to make your apps Y2K ready

IBM

1. **Field expansion**
   - **The best, more later**
2. **Century window**
   - **Temporary, not too bad, more later**
   - **Sliding or fixed window**
3. **Century encoding/compression**
   - **Can't read in dumps, listings**
   - **Must translate whenever externalized**
   - **Translation could be wrong**
   - **Maintenance more difficult**
4. **Integer format dates**
   - **Can't read in dumps, listings**
   - **Must translate whenever externalized**
   - **Translation could be wrong**

More on integer format dates

- Provided by COBOL Intrinsic Functions and by Language Environment date/time callable services
- Number of days from some point in the past
- PRO: Use only 4 bytes to store a date
- CON: Too many Integer format starting dates:
  - CICS, MVS, DB2 - start with Jan 1, 1900
  - PL/I - no integer date values
  - C - start with Jan 1, 1970
  - COBOL - start with Jan 1, 1601
  - LE/370 - start with Oct 15, 1582
- Solution for COBOL: Use COBOL for MVS & VM Install option
  - INTDATE(LILIAN) or INTDATE(ANSI)
  - Use INTDATE(LILIAN) to get LE/370 Lilian dates from Intrinsic Functions

Long-term solution: Field Expansion

- You must eventually:

  1. Re-write applications

     - Use COBOL Intrinsic Functions to get 4-digit year dates OR

     - Use LE/370 callable services to get 4-digit year dates

  2. AND Rebuild databases and files with 4-digit years

- The above are the long-term solutions that will take you and your company through to the year 9999

- However, there is a short-term solution that is easier

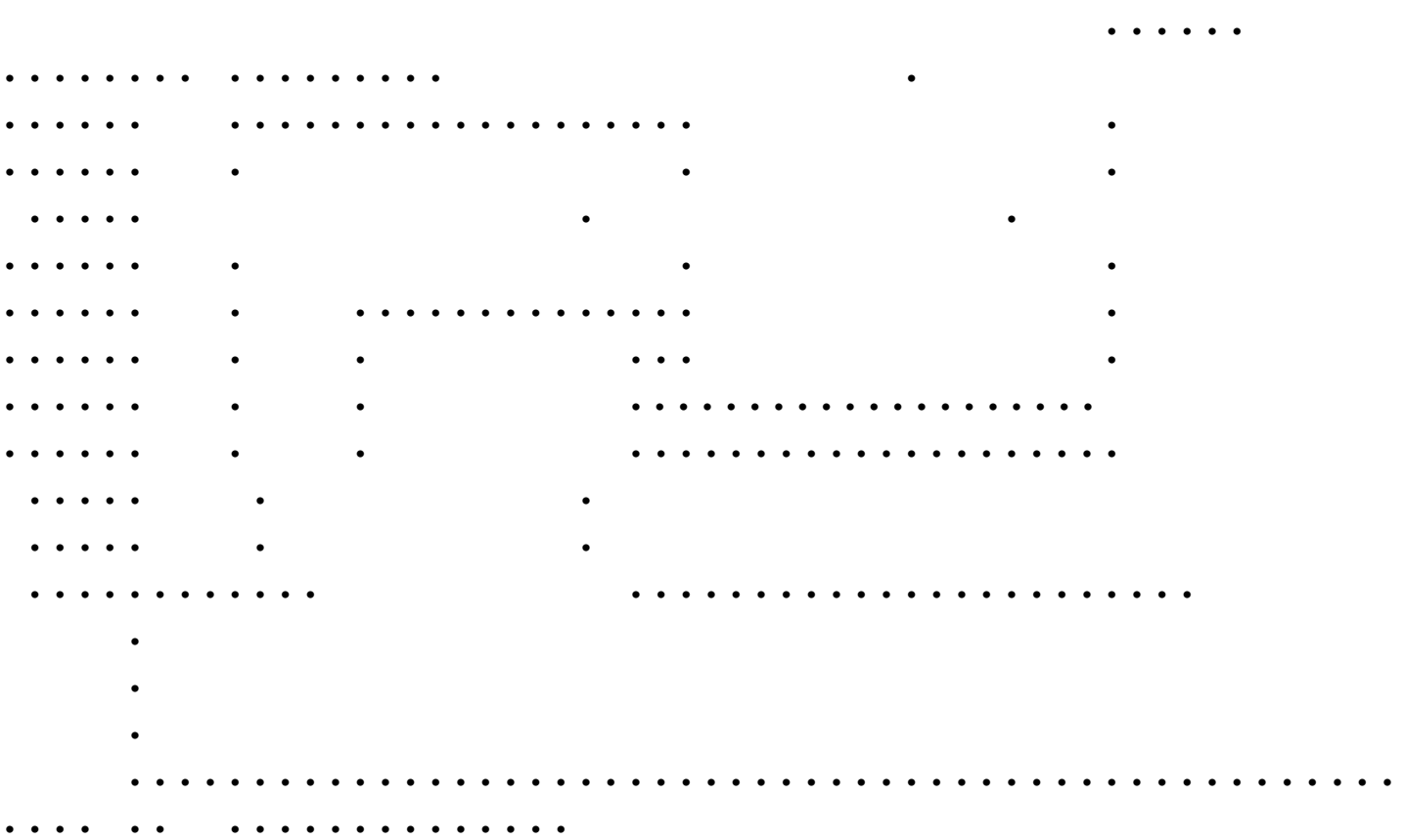
Short-Term solution: Century window

- Use the LE/370 century window in old applications to interpret 2-digit year dates as 4-digit year dates

- Advantages:
  - Fewer program changes, no database change
  - Cheaper, quicker, easier

- Disadvantages:
  - Performance will be slower with the increased logic
  - This only buys you a few years, depending on the application
  - You still MUST change all date programs and databases eventually to use 4-digit years

Short-Term vs long-term: which is better?

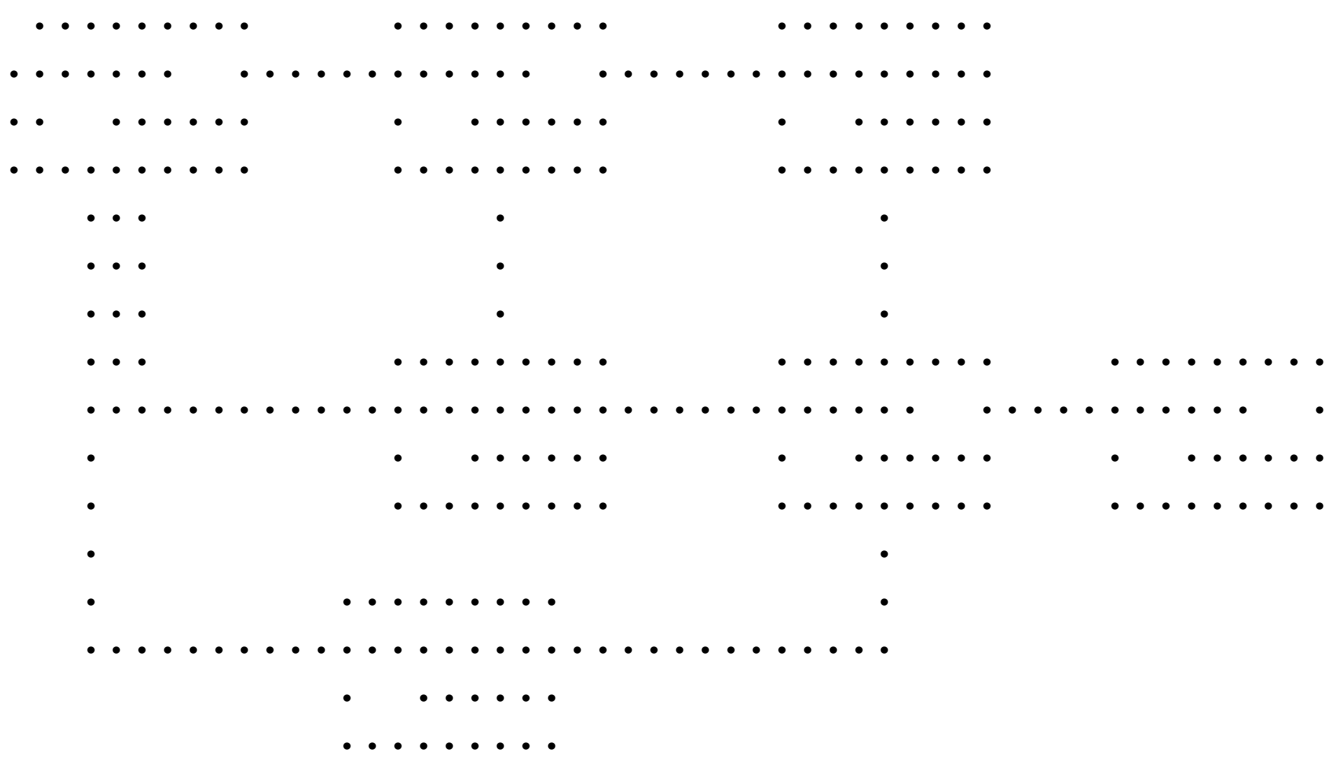


- Not only will it not work forever:
  - It will cost more to maintain
  - You will have to do it all again later!

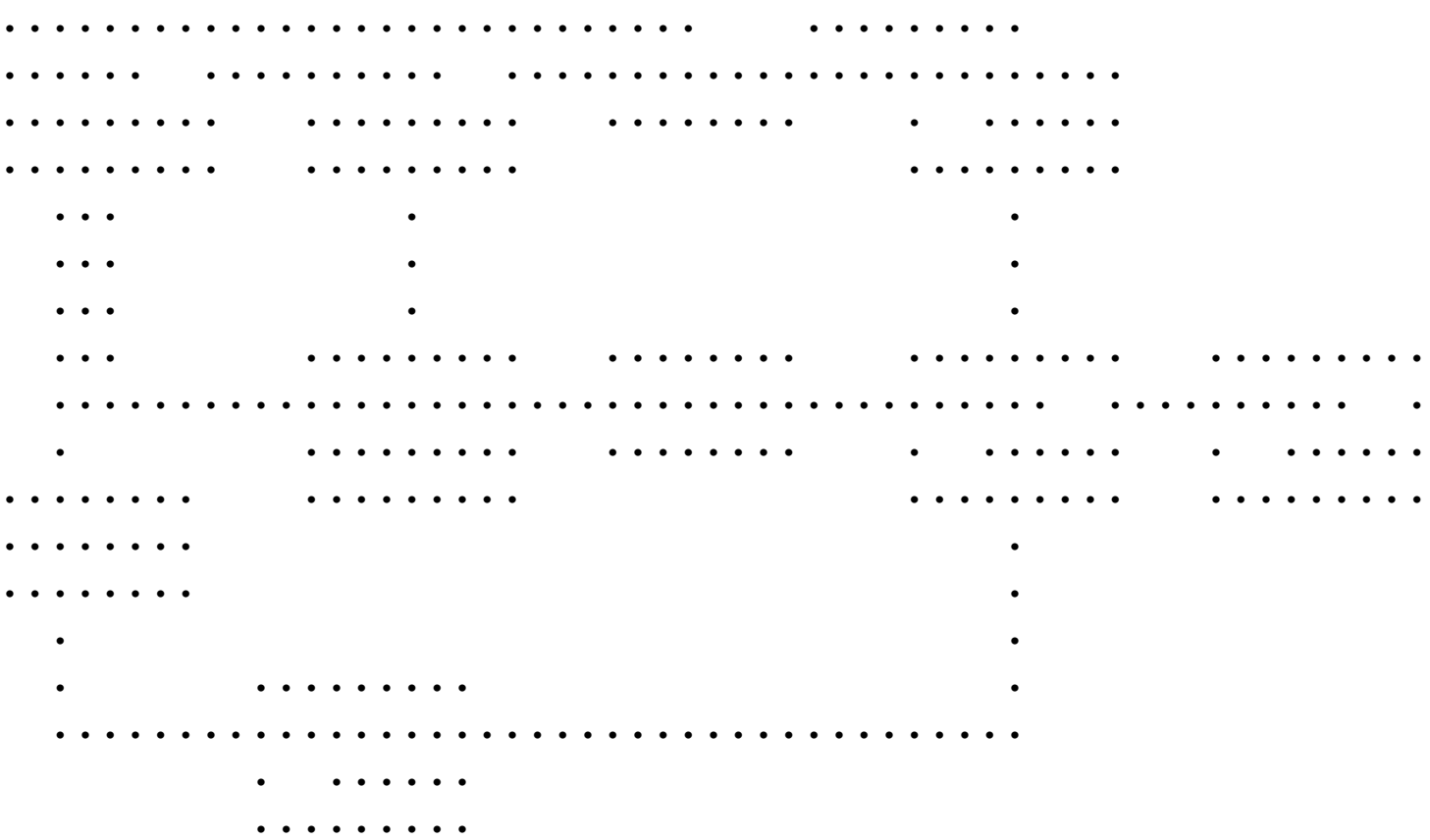
What to do with historical data with field expansion method?

- Basically just two choices for data:
  - Expand all old files/databases
  - Keep copies of old data
- Which leads to choices for code:
  - Copies of old code to access the data as well or
  - Code needs to handle both 2-digit and 4-digit years
    - Could use conditional ″bridges″ when accessing old data

Partitions and bridging



Partitions and bridging



- Only changed APP1, APP2, DATA1

- Burn bridges after DATA or APPS changed

DFSORT and the century window

IBM

- Apply fix for APAR PN71337 (PTF UN90139) to DFSORT R13

- To use the century window function in a COBOL SORT you need to tell DFSORT when your century window starts and which fields are to be sorted using the century window

- Use the following 2 control statements in JCL:

  1. OPTION Y2PAST
     - OPTION Y2PAST=nn
       - Where nn=number of years back to start of window
     - OPTION Y2PAST=nnnn
       - Where nnnn=fixed window start date
  2. SORT FIELDS=(n,n,Y2C,A,...)
     - Where Y2C, Y2Z, Y2P, and Y2D identify 2-digit year fields in character, zoned decimal, packed decimal, and decimal form

- For more information, see DFSORT documentation

www.storage.ibm.com/storage/software/sort/srtmhome/htm

DFSORT and the century window

IBM

- The OPTION statement can be coded in the IGZSRTCD DD statement

```
.......... ..     .
       ...... ..........
```

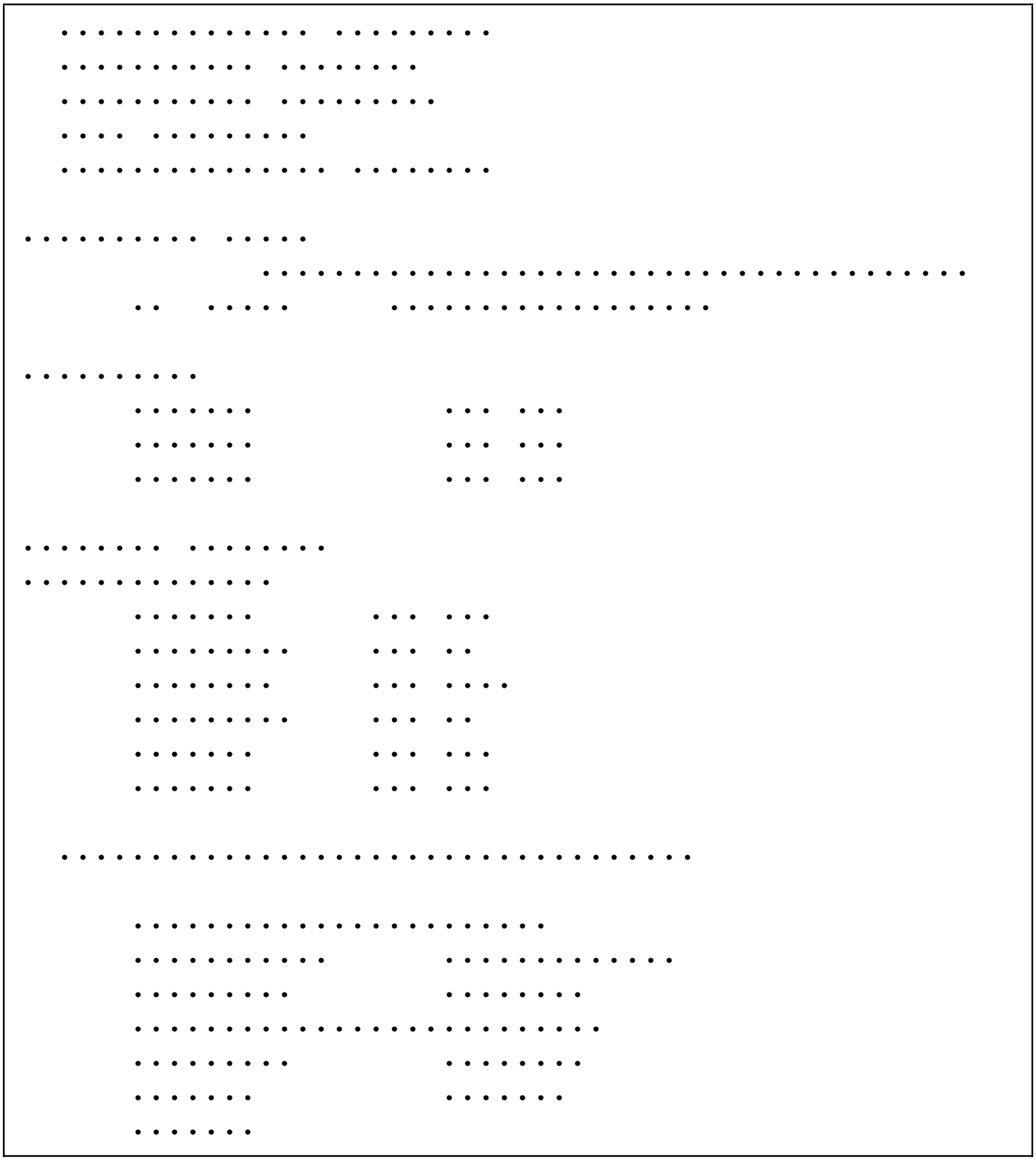
- The SORT control statement must be coded in a DFSPARM DD statement

- To SORT records that start with mm/dd/yy as yyyymmdd:

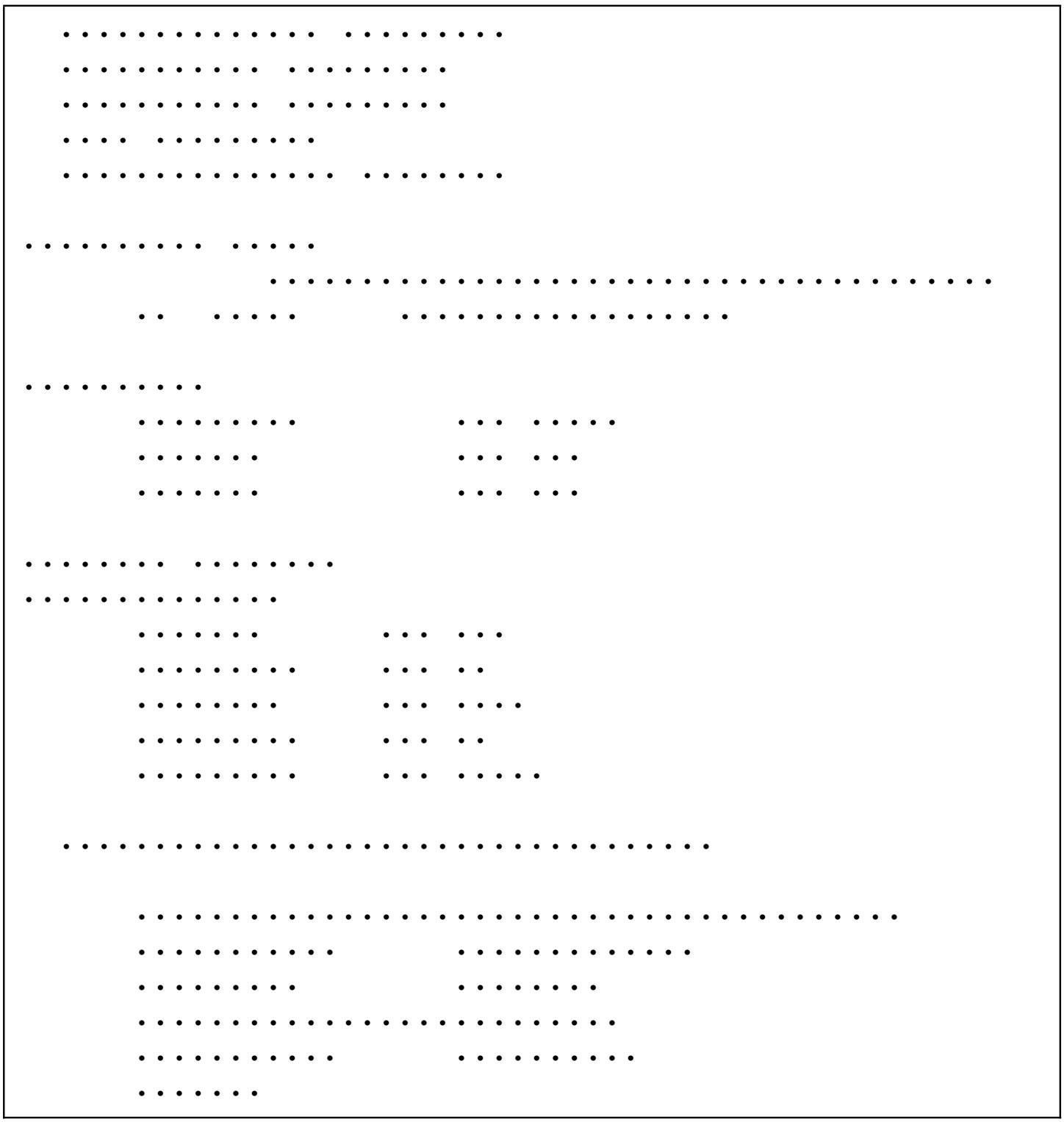
```
.........  ..     .
       ........................  ......................
                    .........   ......................
                    .........   ......................
```

- Other notes on SORT:
  - Use a local name if your installation changed the default name DFSPARM
  - A SORT statement can not be specified in IGZSRTCD
  - This will override the SORT FIELDS produced by the compiler, so you must make sure that you identify all fields that are to be sorted.
  - These are the ASCENDING/DESCENDING KEYS in the SORT statements in your COBOL program

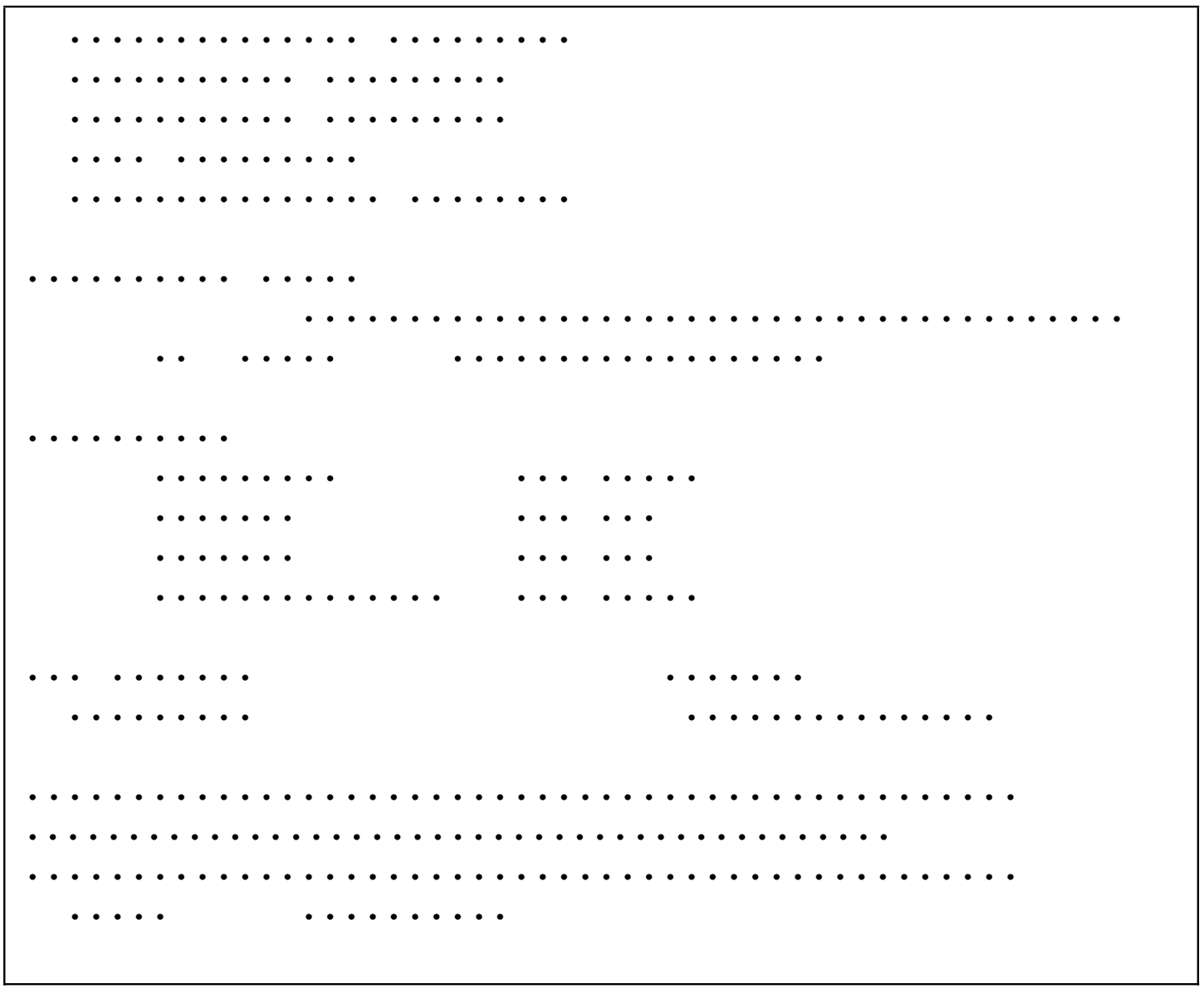
COBOL program samples



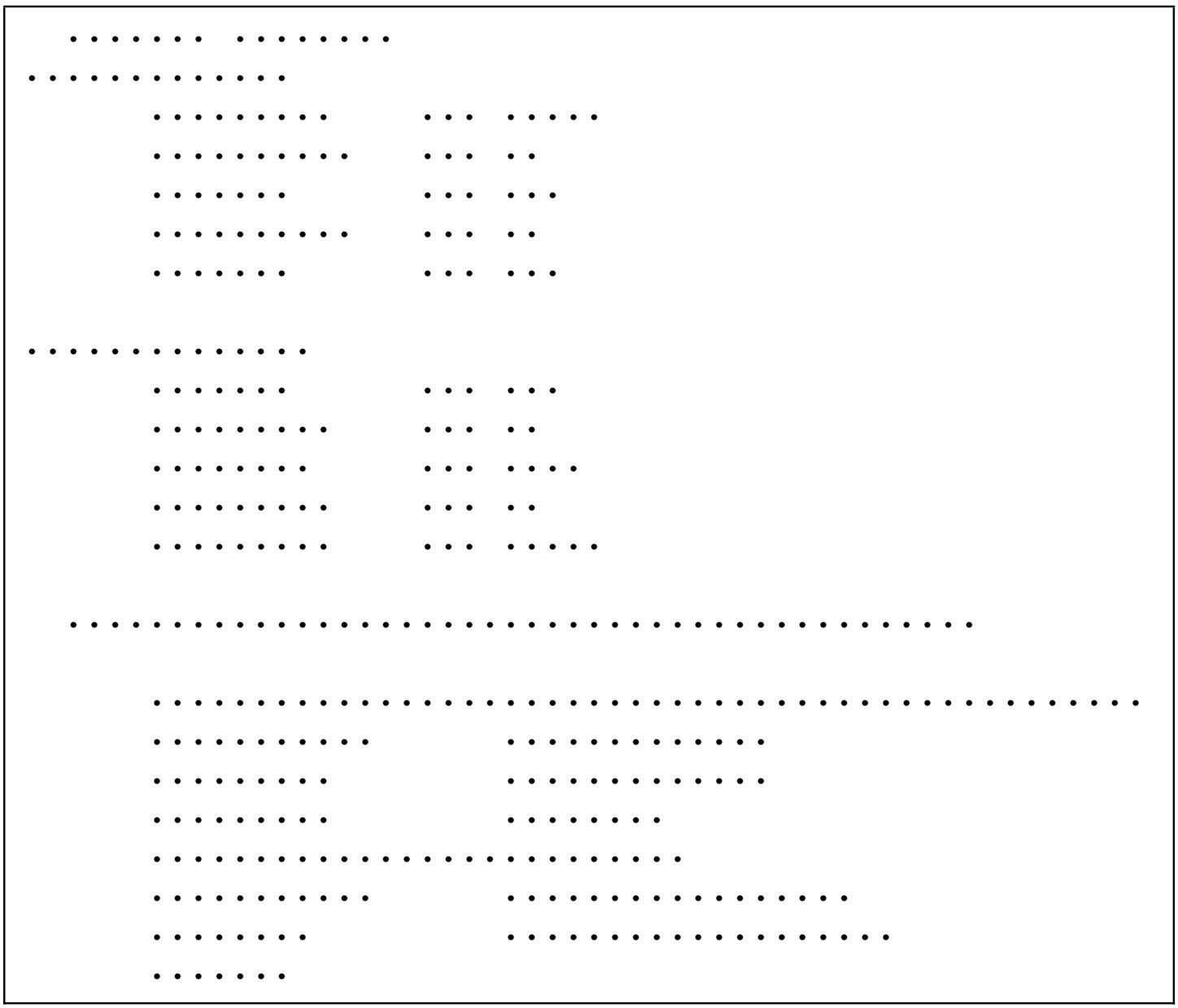
COBOL program samples



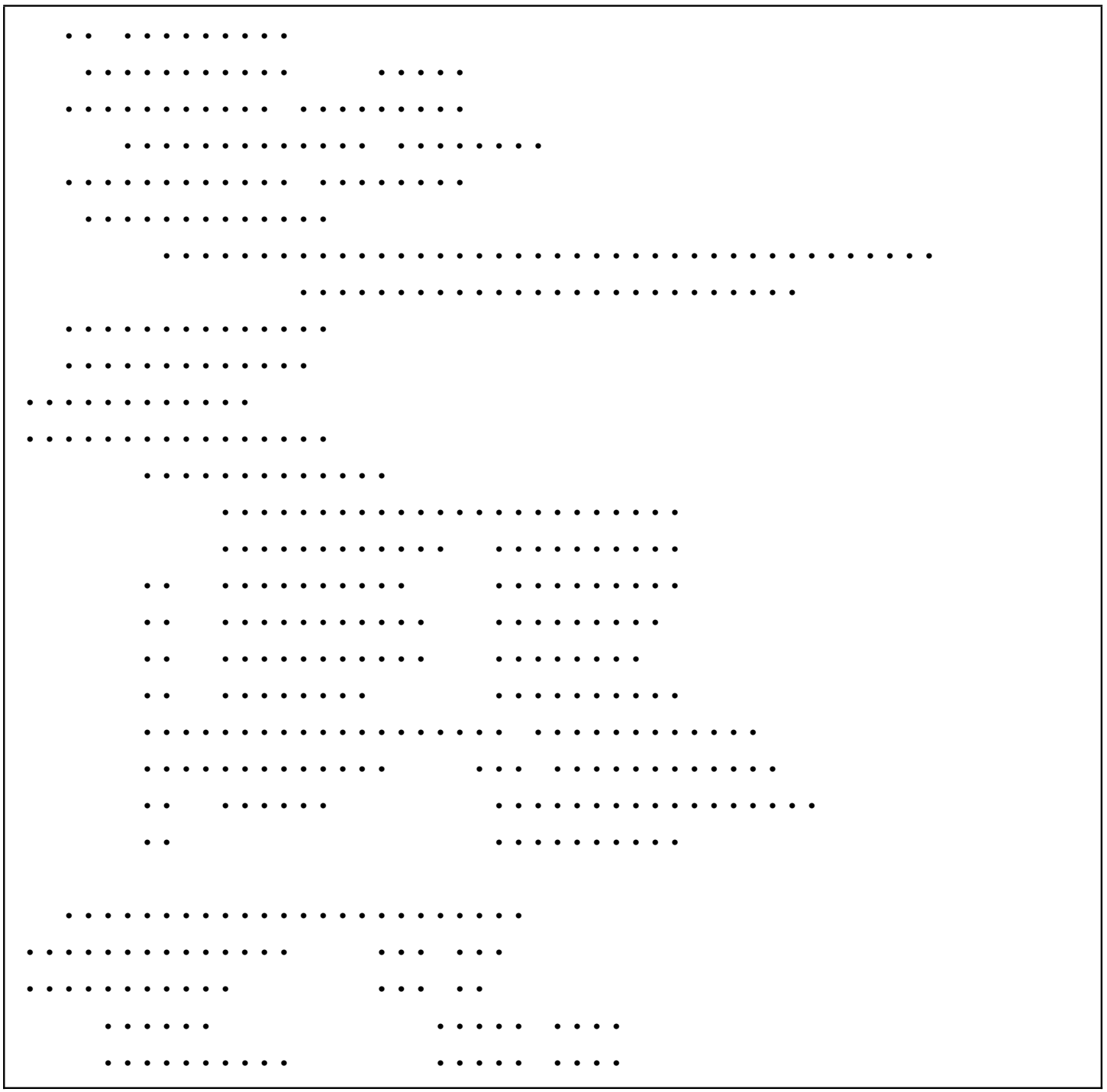
COBOL program samples



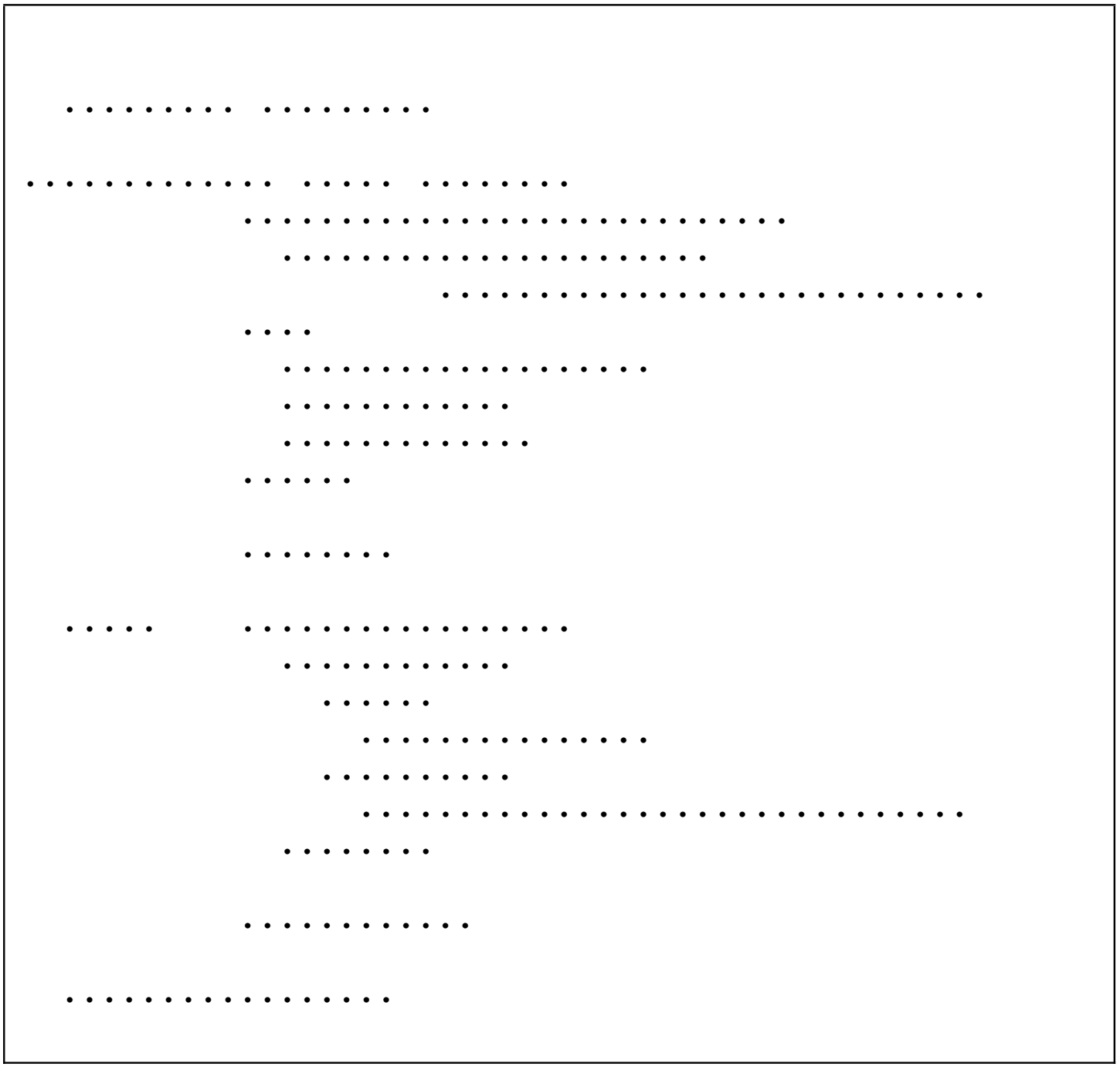
COBOL program samples



COBOL example MAIN

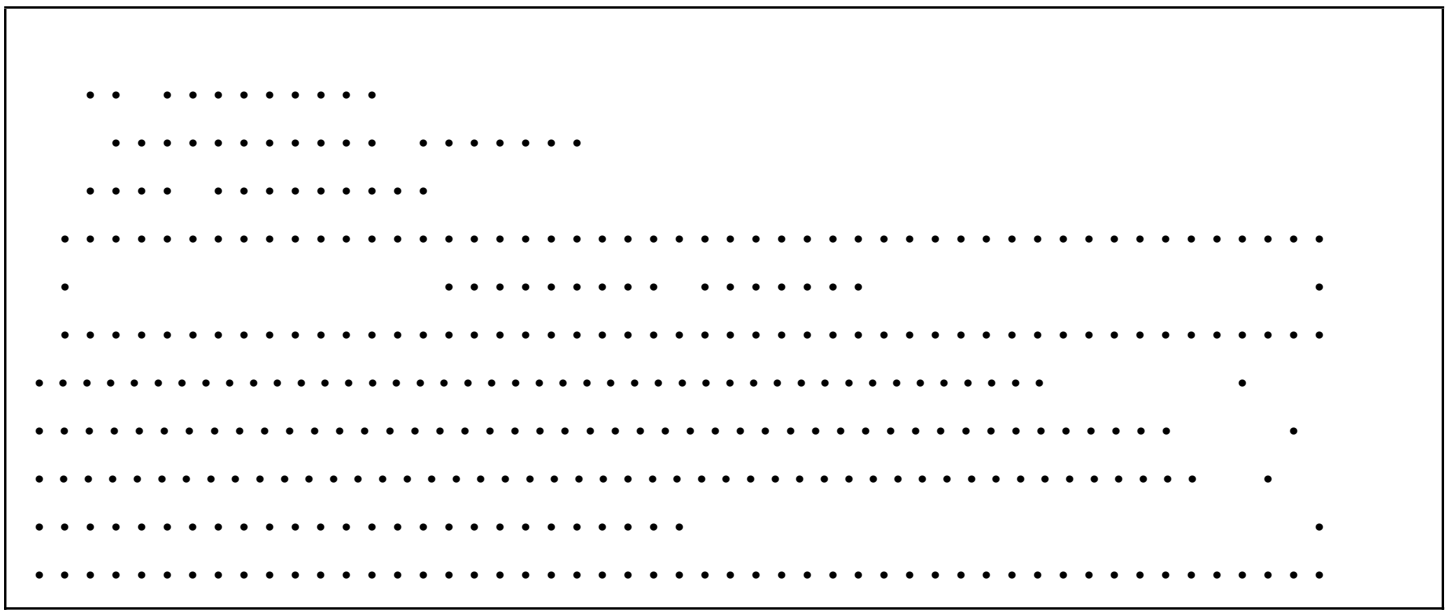


COBOL example MAIN

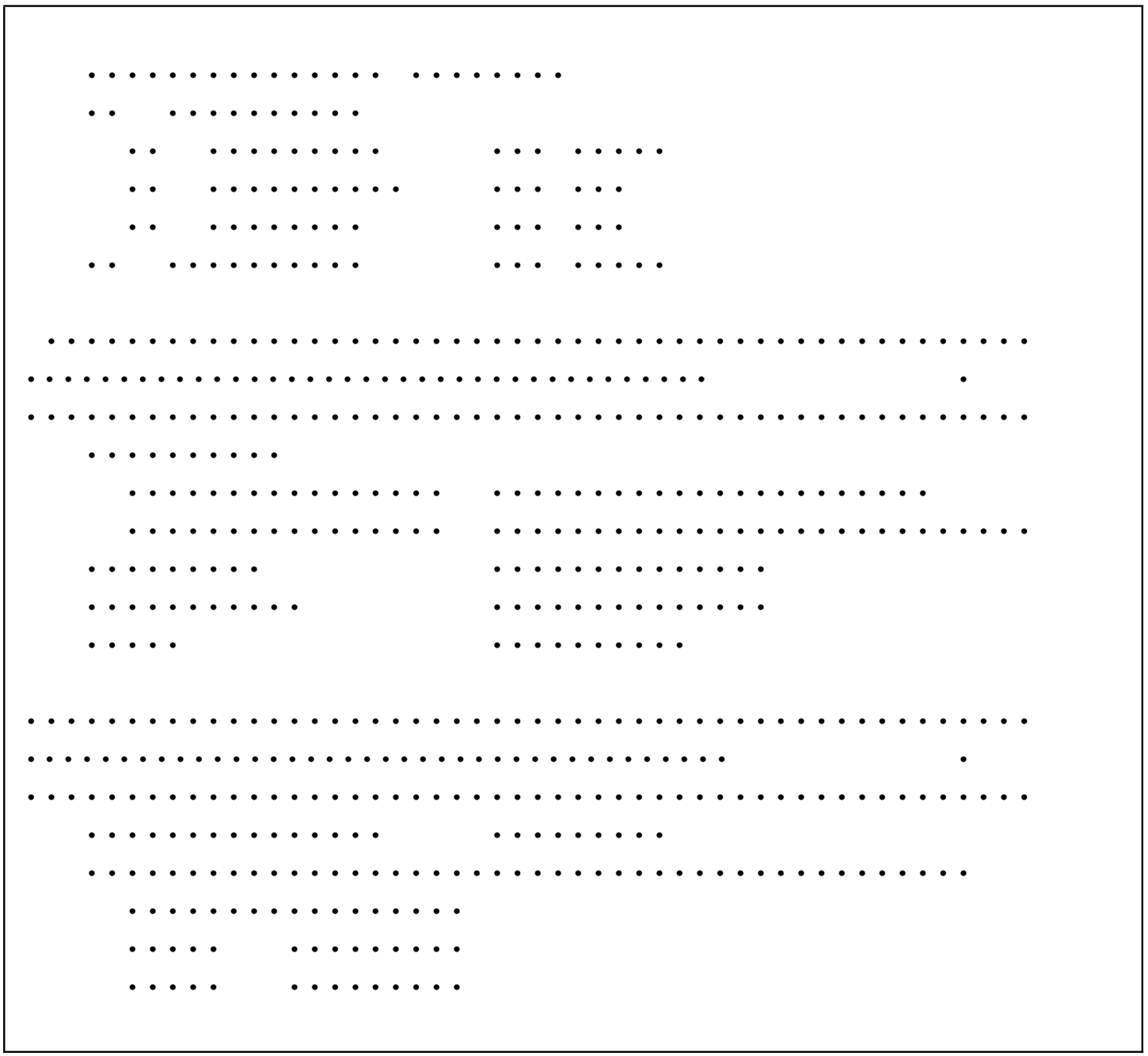


COBOL example SUBRTN

IBM

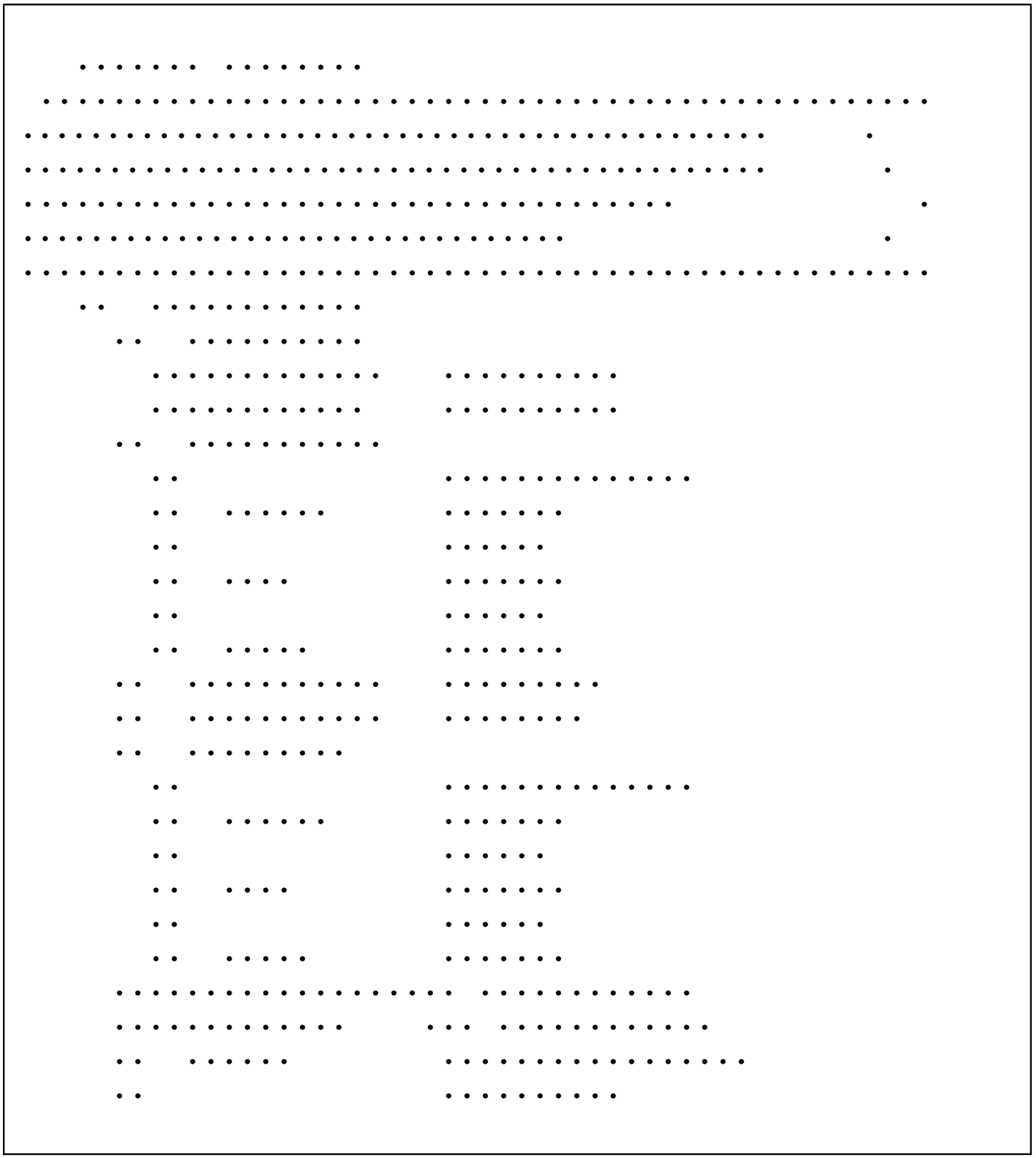


COBOL example SUBRTN



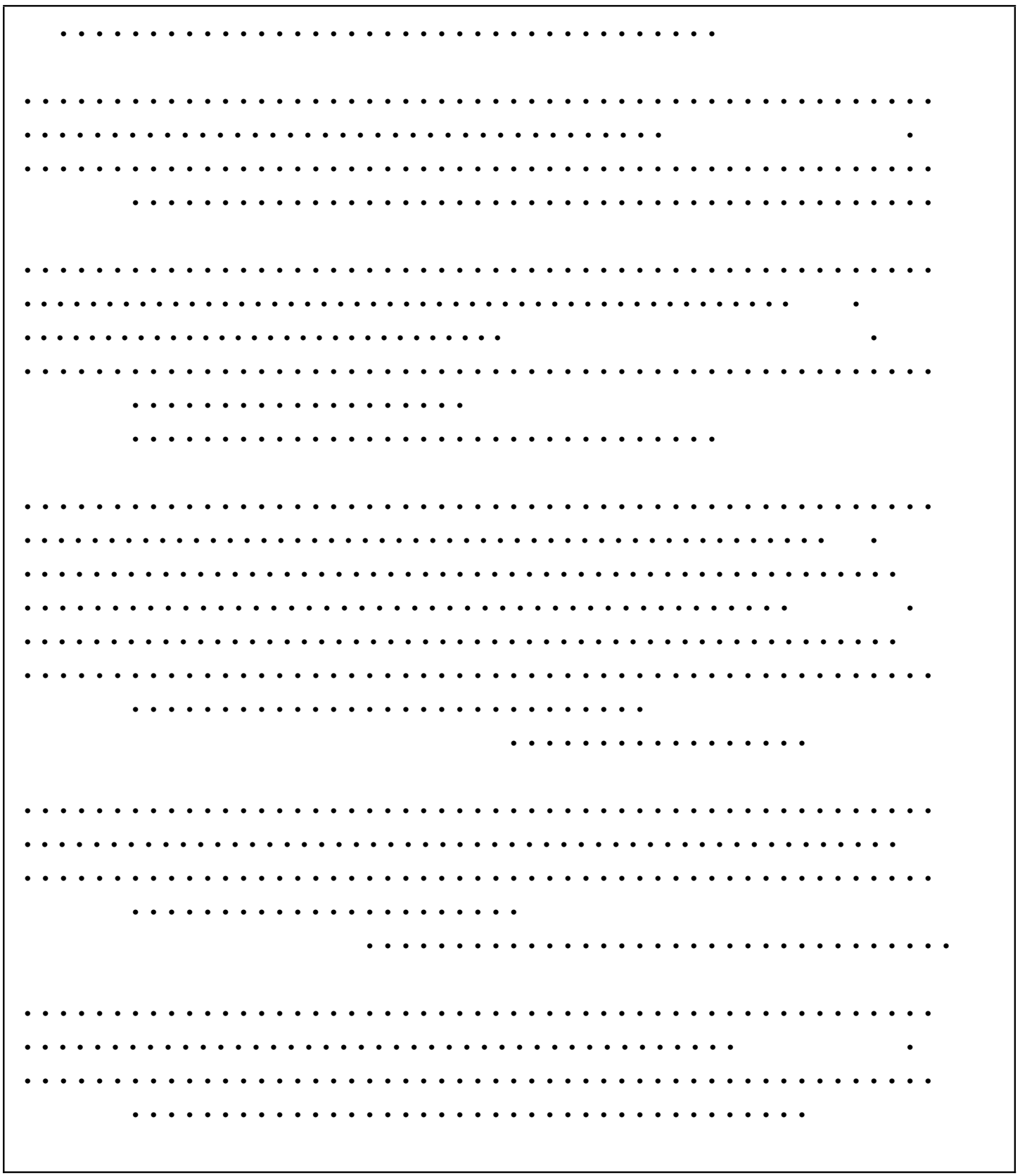
COBOL example SUBRTN

IBM



COBOL example SUBRTN

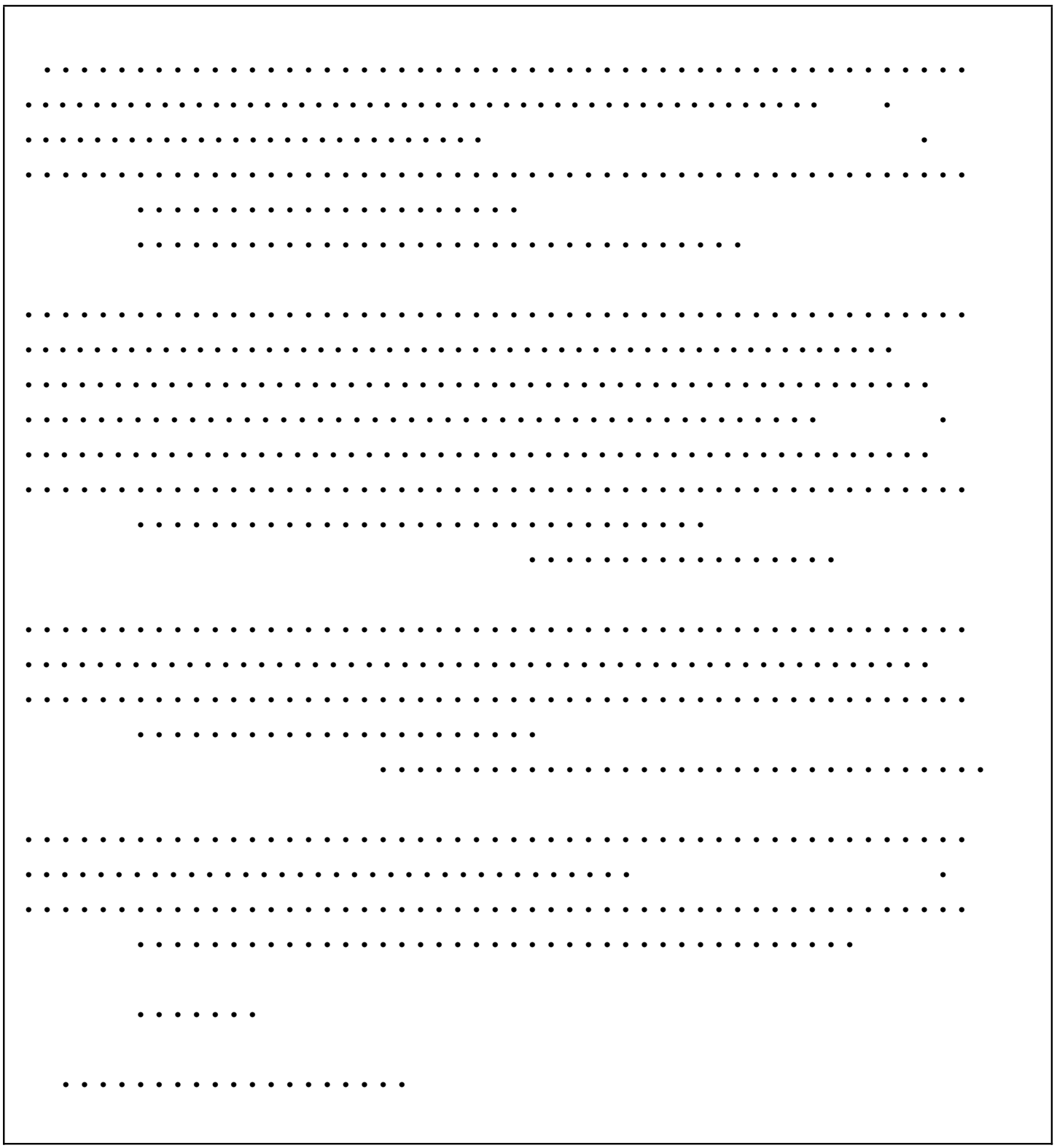
IBM



COBOL example SUBRTN

IBM

COBOL example SUBRTN



Testing your applications

IBM

- Regression test
  - Current production data
  - Regular test environment
- "Time Machine" test
  - Keep future date machines separate from others
  - Do not allow future dates to corrupt:
    - Networks
    - Datasets
  - Ideal "Time Machine": PC Server 500 System/390
- Other future test
  - Data with future dates

Look out!

IBM

- 25% of companies will go out of business in the year 2000
  - Don't let yours be one of them
- Make sure your software vendors are fixing their Year 2000 problems
  - If not, you need to plan for replacements!
- Make sure all of your companies suppliers are also fixing their Year 2000 problems
  - Remember the GM brake division strike?

How to get more information

IBM

- **GC28-1251: The Year 2000 and 2-digit dates:**

  A Guide For Planning and Implementation

  - **The manual can be downloaded from the IBM Year 2000 website**
  - **http://www.software.ibm.com/year2000/index.html**

- **Year 2000 Technical Support Center**

  Ask questions via:

  - **YEAR2000 CFORUM**
    - **(Available though TALKLINK of IBMLINK)**
  - **Y2KTSC@vnet.ibm.com**

- **A non-IBM Year 2000 web site (de Jaeger)**

  - **http://www.year2000.com**