Multimedia in Teaching of Mathematics and Informatics

Prof. dr Miroslav Ćirić

Faculty of Sciences and Mathematics

University of Niš, Serbia & Montenegro

Introductory Remarks

- **This lecture is based on**
  - **my paper "LaTeX Tools for Web Publishing, Screen Presentations and Electronic Examinations" published in the Proceedings on the Workshops on Computer Science Education;**
  - **my lecture given for elementary and middle school teachers of mathematics and informatics (January, 2004);**
  - **my lectures on the course on Innovations in teaching of mathematics and informatics, for teachers and teaching assistants from Department of Mathematics and Informatics, Faculty of Sciences and Mathematics, University of Niš (January – March, 2004)**

Why TEX?

- Most people place text on a computer and arrange material on a page with a word processor.
- Word processors are easy to begin with.
  - To get a blank line between two paragraphs we enter it in.
  - to make a reference to the bibliography we type it into the text in style that we need, etc.
- It seems simple.

- But, a word processor will suit our needs only if our documents are brief, short to medium sized, structurally simple, and entered by hand.
- As the document gets to be a bigger and tougher job, laying it out ourselves becomes a problem.
  - For example, in a document with hundreds of bibliographic entries we can not be sure that all the entries are formatted in the same way.

- But, a word processor will suit our needs only if our documents are brief, short to medium sized, structurally simple, and entered by hand.

- As the document gets to be a bigger and tougher job, laying it out ourselves becomes a problem.
  - For example, in a document with hundreds of bibliographic entries we can not be sure that all the entries are formatted in the same way.

- Even bigger problem appears when we have to typeset technical material containing a lot of mathematical symbols and complex mathematical formulas.

- **But, a word processor will suit our needs only if our documents are brief, short to medium sized, structurally simple, and entered by hand.**
- **As the document gets to be a bigger and tougher job, laying it out ourselves becomes a problem.**
  - **For example, in a document with hundreds of bibliographic entries we can not be sure that all the entries are formatted in the same way.**
- **Even bigger problem appears when we have to typeset technical material containing a lot of mathematical symbols and complex mathematical formulas.**
- **In such cases, the best solution is to use TEX.**

About TEX

- TEX is a system (program) for computer typesetting and text processing
  - it is primary devoted for texts containing a lot of mathematical symbols and complex mathematical formulas.
  - TEX's creator: Donald Knuth (Stanford University), 1978.

About TEX

- **TEX is a system (program) for computer typesetting and text processing**
  - it is primary devoted for texts containing a lot of mathematical symbols and complex mathematical formulas.
  - TEX's creator: Donald Knuth (Stanford University), 1978.
- **TEX is a programming language**
  - TEX is a macro language – "simple" TEX commands are defined in input files through complex macro commands.
  - TEX is a markup language (very similar to HTML and XML).

Advantages of TeX

- TeX has the best output
  - TeX possesses much better algorithms for line/page breaking than other text processors.
  - This fact is not surprising if we keep in mind that its creator is the world leading expert in design of algorithms.
  - TeX has more readable output than other text processors.
    - Spaces between words are unified.
    - TeX has an excellent hyphenation.

- TEX's output quality is fully expressed when we typesset text containing mathematical symbols and formulas

- TEX's output quality is fully expressed when we typesset text containing mathematical symbols and formulas
  - TEX automatically classiffies each mathematical symbol as a variable, relation or operation symbol, etc. and sets them with appropriate amounts of surrounding space

$$x, y, \ldots \qquad x + yz \qquad x \leq y \qquad \cos x$$

- **TEX's output quality is fully expressed when we typesset text containing mathematical symbols and formulas**
  - TEX automatically classiffies each mathematical symbol as a variable, relation or operation symbol, etc. and sets them with appropriate amounts of surrounding space

    $$x, y, \ldots \qquad x + yz \qquad x \leq y \qquad \cos x$$

  - TEX automatically sizes subscripts, superscripts, and many other things, with respect to their level in the formula

    $$x_1 \qquad x_{n_1} \qquad x^n \qquad x^{2^n} \qquad x^{n_1} \qquad \frac{\frac{n}{2-n^2}+3}{n^2-\frac{1-n}{1+n}}$$

- TEX automatically sizes parentheses, root, sum, product, or integral signs and other, with respect to the size of the formula on which they act

$$2(x^2+1) \quad \left(\frac{x+1}{2x-3}-\frac{1}{x^2-1}\right) \quad \sqrt{x^2-1} \quad \sqrt{\frac{x+1}{2x-3}}$$

- TEX automatically sizes parentheses, root, sum, product, or integral signs and other, with respect to the size of the formula on which they act

$$2(x^2+1) \qquad \left(\frac{x+1}{2x-3}-\frac{1}{x^2-1}\right) \qquad \sqrt{x^2-1} \quad \sqrt{\frac{x+1}{2x-3}}$$

- TEX distinguish between mathematical expressions in the text mode, such as, for example, $\sum_{k=1}^{n}\frac{1}{k^2-1}$

- TEX automatically sizes parentheses, root, sum, product, or integral signs and other, with respect to the size of the formula on which they act

$$2(x^2+1) \qquad \left(\frac{x+1}{2x-3}-\frac{1}{x^2-1}\right) \qquad \sqrt{x^2-1} \quad \sqrt{\frac{x+1}{2x-3}}$$

- TEX distinguish between mathematical expressions in the text mode, such as, for example, $\sum_{k=1}^{n}\frac{1}{k^2-1}$, and the display mode

$$\sum_{k=1}^{n}\frac{1}{k^2-1}$$

Superior Engineering

- TEX is very fast, considerably faster than other text processors.
- TEX is less computer's memory and disk space consuming.
- TEX is stable – it is in wide use, with a long history. It has been tested by millions of users on demanding input, and it has never eaten any document.
- TEX is stable but not rigid, it is extendible, so that innovations can be easily added on.
- It is especially open for adding new macro packages and document templates, the so-called document classes.
- The best known innovations are LATEX, AMS-TEX and others.

➠ TEX documents are plain text files

- TEX documents are compact, much smaller than documents in the binary format, as other text processors create.
- TEX documents are more suitable for transfer (by e-mail, for example) than documents in the binary format.
- TEX documents are portable – it can be used on various computing platforms, under diverse operating systems.
- TEX documents can be easily generated automatically, for example, from a database, XML document or something similar.

- **TEX documents are plain text files**
  - TEX documents are compact, much smaller than documents in the binary format, as other text processors create.
  - TEX documents are more suitable for transfer (by e-mail, for example) than documents in the binary format.
  - TEX documents are portable – it can be used on various computing platforms, under diverse operating systems.
  - TEX documents can be easily generated automatically, for example, from a database, XML document or something similar.

- **TEX is a free and open software**
  - The majority of TEX's distributionm packages is free – for example, MiKTEX, which is in the widespread use.
  - Source codes of TEX and its main components are open.

➠ TEX became a standard

- TEX became a standard for preparation of scientific and technical publications.
- As a result, the majority of publishers of technical material are set up to work with it – they require manuscripts to be prepared by TEX.
- TEX is directly supported by other technical software. For example, most computer algebra systems, such as Maple and Mathematica, give output in TEX.
- No doubt any technical software developed in the future will support TEX, too.

- **TeX output can be anything**
  - TeX produces a DVI file (abbreviation for ”device independent”), as its primary output format.
  - But, the TeX engine’s results can be easily converted to
    - a printer language, such as PostScript,
    - a web language, such as PDF, HTML or XML,
    - or, probably, to whatever will appear in the future.
  - And, the typesetting (line breaks, etc.) will be the same no matter where our output appears.

About LaTeX

- LaTeX is an extension of TeX created by Leslie Lamport, 1984.
- **The main idea of LaTeX**
  - autors should concentrate one's attention on the logical structure (markup) of the text, rather than to the formating details.
  - LaTeX encourages separation of formating details and contents.
  - Formating details, from one side, and text, from another side, are stored in separate files.
  - As a resuls, formating details will be uniform trough the whole text.

- Other important characteristics of LaTeX
  - automatic numbering,
  - cross-references,
  - and much more.
- LaTeX itself is extensible
  - There are thousands of ”style files”, which do everything from adapting the basics to the needs of the American Mathematical Society (AMS-LaTeX), to making cross-references into hyper-references (hyperref), etc.

TEX and PDF

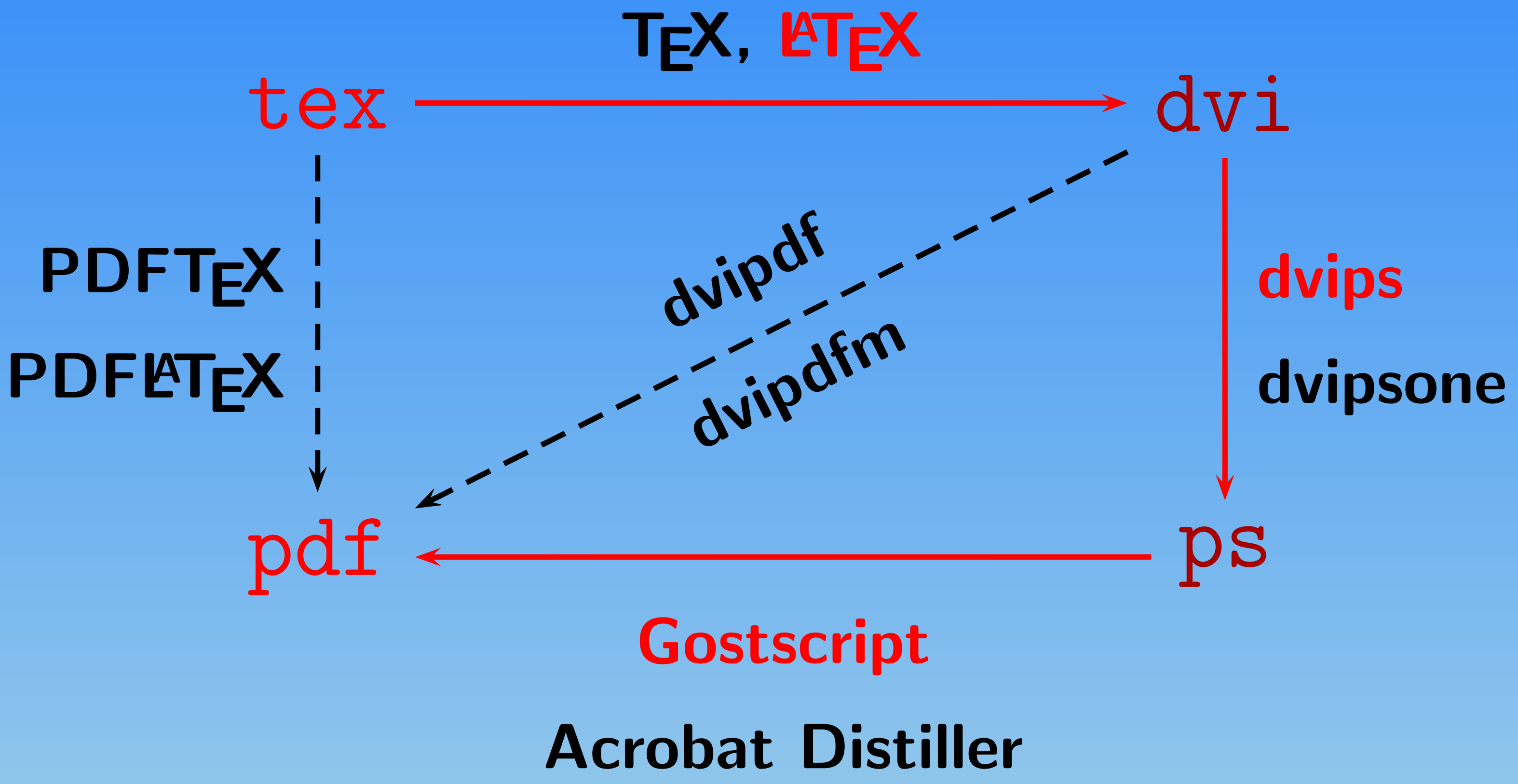
- **TEX produces a DVI file, as its primary output format**
  - DVI file can be printed directly given an appropriate printer driver or viewed on the screen using some of many existing viewers.
  - Although DVI means “device independent”, DVI format is not completely independent.
  - For example, graphics is not incorporated into the DVI file – it is called from outside when the DVI file is viewed or printed.
  - This endangers portability of documents in the DVI format.
  - DVI supports color, but not other up-to-date multimedia elements, hyperlinks and other.

- DVI file can be easily converted into PostScript
  - Recall that PostScript is a page description language created in the Adobe Systems company.
  - Converting DVI into PostScript we solve the portability problem, because everything, including graphics, is incorporated into the PostScript file.
  - From that reason, documents created by TEX are often distributed in the PostScript format.
  - But, PostScript does not support multimedia, hyperlinks and similar things.

- **Solution of all problems – output in the PDF format**
  - **PDF** is a file format developed also in the Adobe Systems company.
  - PDF is an abreviation for “Portable Document Format”
  - This means that the docunemnts in PDF format are represent in a way which does not depend on the application software, hardware, and operating system used for its creation.
  - PDF is a simplified version of the PostScript language, aimed far more at quality on-screen display, document exchange, and hypertextual applications, than at printing quality.

- PDF has a very efficient font-embedding/replacement system which allow fonts to travel with the documents.
- PDF posesses a structured storage system to bundle these elements into a single file, with data compression where appropriate.
- In addition to that, PDF is extended to enable to
  - use hyperlinks,
  - use various dynamic effects for page transitions,
  - work with forms, call external applications, sound and video,
  - embed JAVAScript programs into a PDF document,
  - and much more.
- This extension is known as pdfmark.

➠ How from TEX to PDF?



tex

dvi

ps

pdf

TEX, LATEX

dvips

dvipsone

Gostscript

Acrobat Distiller

PDFTEX

PDFLATEX

dvipdf

dvipdfm

TEX and PDF in Teaching of Mathematics and Informatics

➠ **Publications**

- paper publications – prepared on a computer and printed on a paper.
- electronic publications – designed to be read from a computer screen and/or printed on a paper.
- screen publications – a special kind of electronic publications designed primarily to be read from computer screen or projected to a movie screen.

➠ **The major points concerning screen design**

(1) Design the text region so that a single page fits on a screen monitor

- ☞ **Having the whole page fit on the screen allows the reader to avoid constant vertical scrolling, that can be distracting and fatiguing when reading large amounts of material.**
- ☞ **Rather than scrolling, it is much easier to simply paginate, go to the next page to continue reading.**

(2) Make the dimensions of the page roughly 3 by 2 (width by height);

(3) Crop the pages to trim off all unnecessary white space around the margins;

☞ The dimensions of 3 by 2 and cropping of white space from around the page will allow the user to magnify the page and the font size to help the eyes read a large amount of text on a screen for long periods of time.

Presentations

- These are electronic documents devoted for projection from a computer to a movie screen, using a multimedia projector.
- Most people use Microsoft PowerPoint, which gives quality presentations supporting all multimedia elements.
- But, PowerPoint is not able to produce quality presentations containing a lot of mathematical symbols formulas.
- The best solution for creating such presentations is to make presentations in the PDF format, using LaTeX.
- Such presentations are high-quality, as the ones created by PowerPoint, even in respect the usage of multimedia elements.

Electronic Testing

- Recently, numerous systems for electronic testing of students appeared.
- The best way for preparation of quizzes containing mathematical symbols and formulas is to use LaTeX, and PDF, as its output format.
- There are several web locations containing quizzes in the PDF format, made by LaTeX.
- The best such system is AcroTeX, created by Donald Story, University of Akron, USA.

PDF presentations

- There is a lot of LaTeX packages for preparation of presentations in PDF format – the so-called “Slide Development Packages”.
- Also, there is a lot of tools (LaTeX packages and programs) which are used to enhance presentations created by some slide development package – they are called “Slide Enhancement Tools”.
- This presentation is prepared using the Seminar package, created by Timothy Van Zandt (Princeton University), and further developed by Denis Girou

Elements of Presentations

- **Backgrounds**;
- **Page Transitions**;
- **Overlays** (displaying page step by step);
- **Navigation Bars and Panels**;
- **Animations**;
- **Movies, Sound, External Applications**.

PSTrics

- High-quality color, graphics (including backgrounds), overlays and animations in PDF presentations are mainly provided by the PSTrics package.
- PSTrics is a collection of TEX macros which give color, graphics, rotation, nodes, trees, overlays and other things, when are translated into the PostScript Language.

Backgrounds

- **Types of Backgrounds Supported by Seminar**
  - **solid backgrounds**;
  - **gradient backgrounds**, based on the “pst-grad” PSTricks package;
  - **improved gradient backgrounds**, based on the “pst-slpe” PSTricks package;
  - **composite backgrounds**, based on PSTricks algorithmic graphics;
  - **backgrounds with external images**.

Solid Background

the default color is “Lavender”

Gradient Background

the starting color is “LightBlue”

the ending color is “NavyBlue”

Gradient Background

with GradientMidPoint = 0.5

(the default is 0)

Gradient Background

with GradientAngle = 45 degrees

Improved Gradient Background

ccslope mode

Improved Gradient Background

radslope mode (the default)

Improved Gradient Background

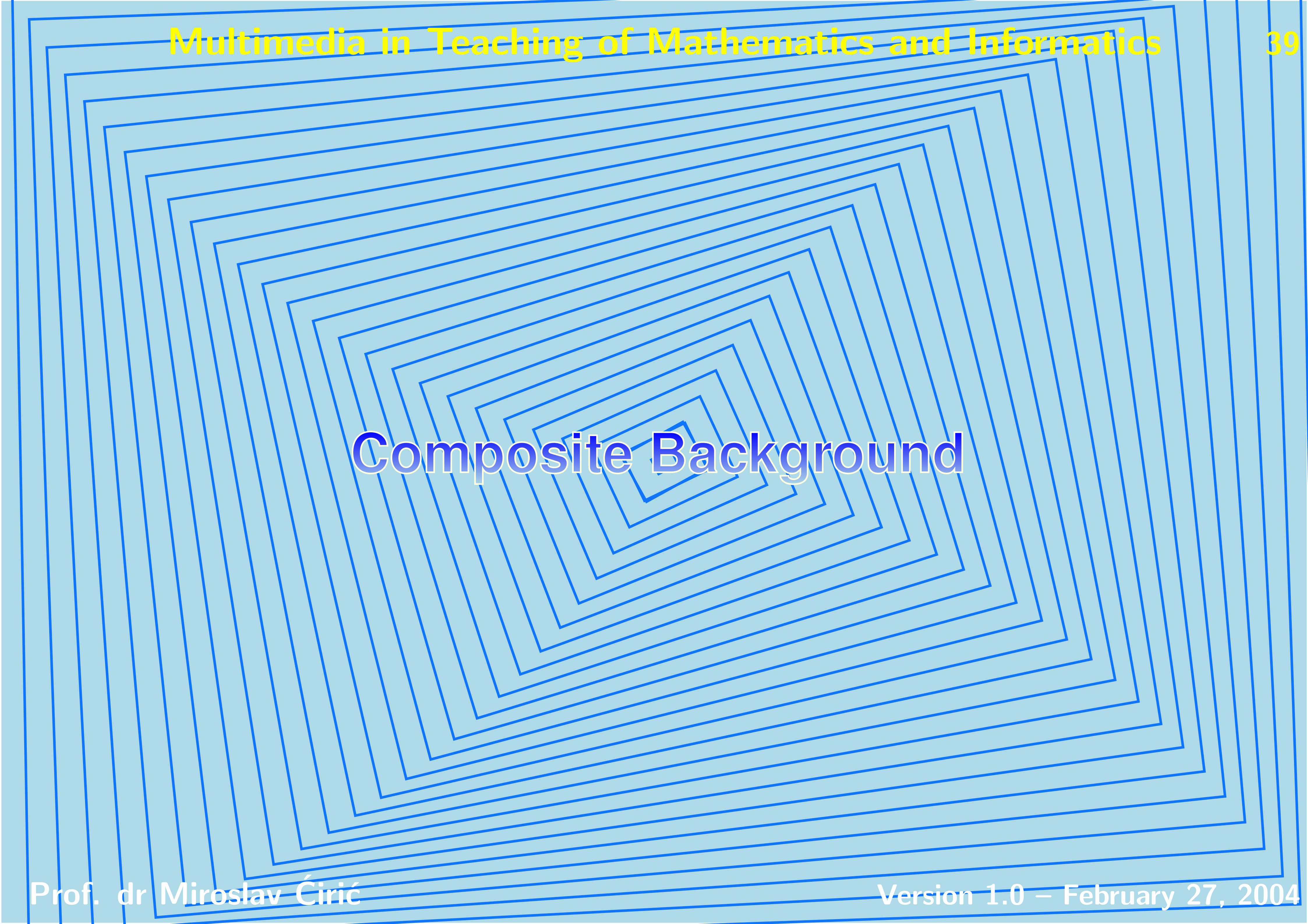
radslope mode (with permuted colors)

Improved Gradient Background

radslope mode, with SlopeRadius = 4

Composite Background

Composite Background



Composite Background

Composite Background

Composite Background

Composite Background

Composite Background

Composite Background

Background with an External Image

Page Transitions

➠ Transition Effects Supported by PDF

Blinds	–	the screen is splitted in several parts to show the new page, like Venetian blinds
Box	–	a box increase or decrease to show the new page
Dissolve	–	the old page is cut in small squares which are progressively replaced by squares of the new one
Glitter	–	like Dissolve, but from one edge to another
Replace	–	the old page is simply replaced by the new one (the default)
Split	–	the screen is splitted in two parts to show the new page
Wipe	–	a line wipes the old page to show the new page

➠ Demonstration of Transitions

Overlays

- **Overlay** is an effect when a page is displayed step by step
- there are two kinds of overlays
  - **Cumulative Overlays**, with each new one added to the preceding.
  - **Progressive Overlays**, where the first slide is kept but each new one will replace the preceding.
- In fact, a PDF presentation means we make a sequence of pages and add some more text on each of them.
- Doing a presentation with Acrobat Reader one can give the effect of dynamically building a page, because pages are updated instantaneously.

An example of cumulative overlays

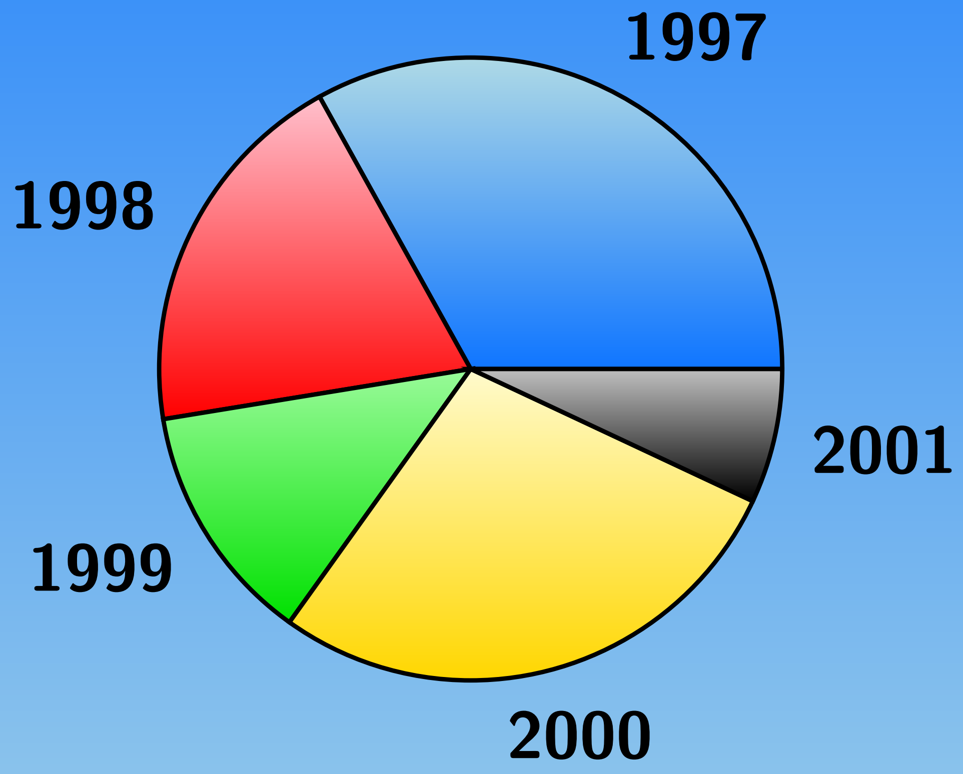


Figure 1: Results of the last five years

An example of cumulative overlays

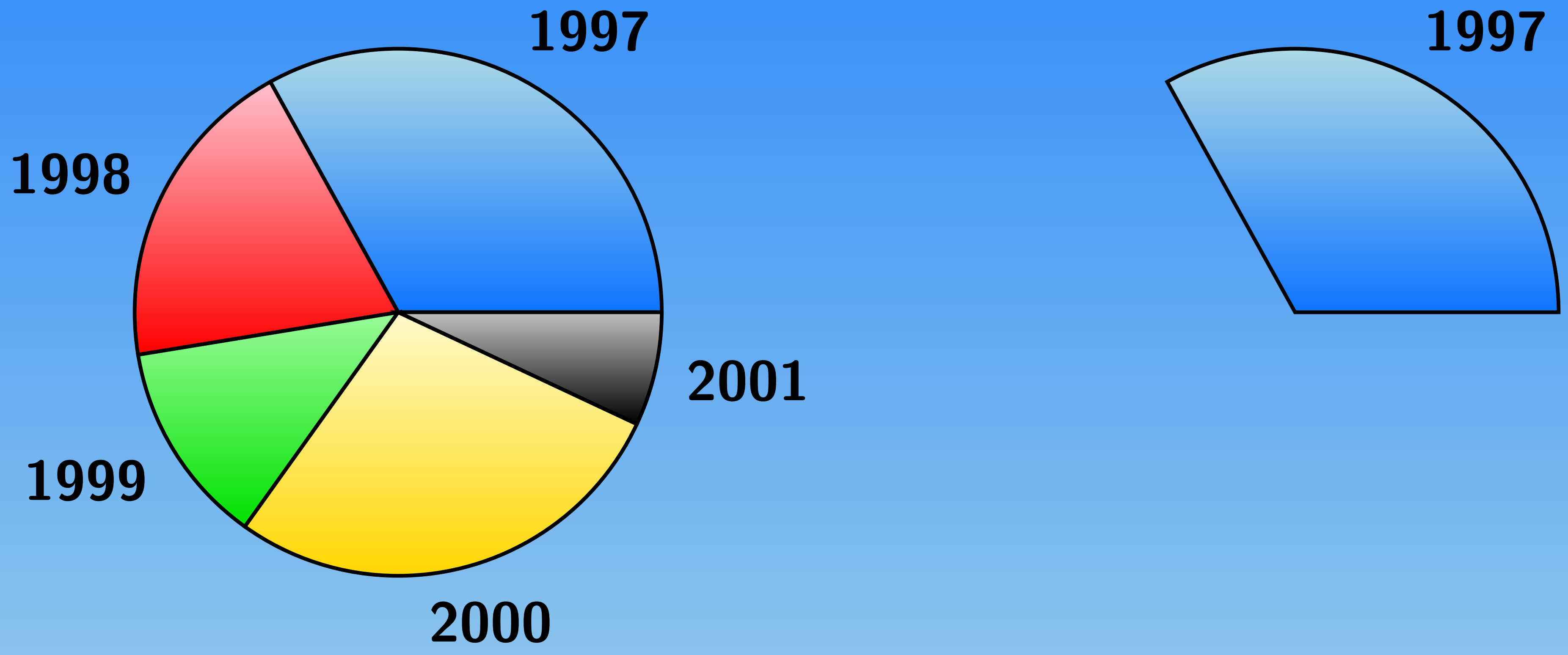


Figure 1: Results of the last five years

An example of cumulative overlays

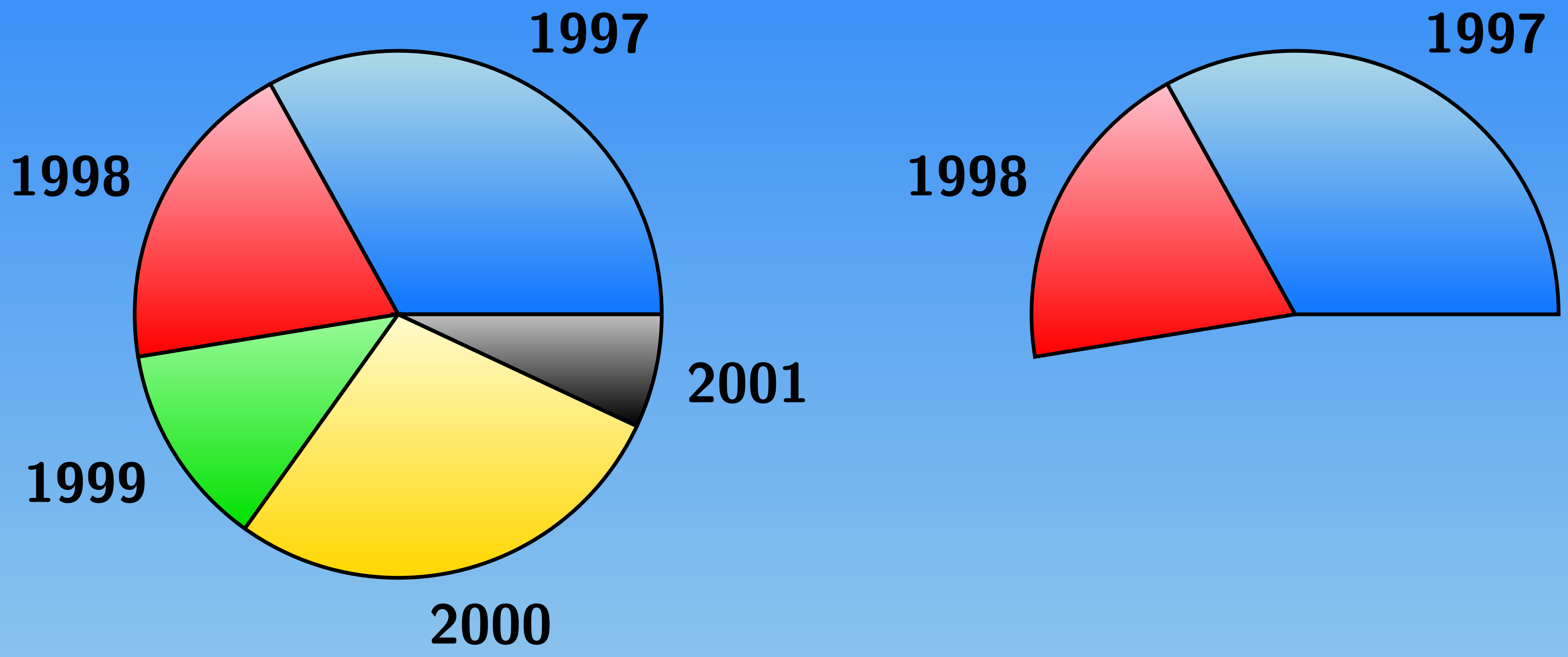


Figure 1: Results of the last five years

An example of cumulative overlays

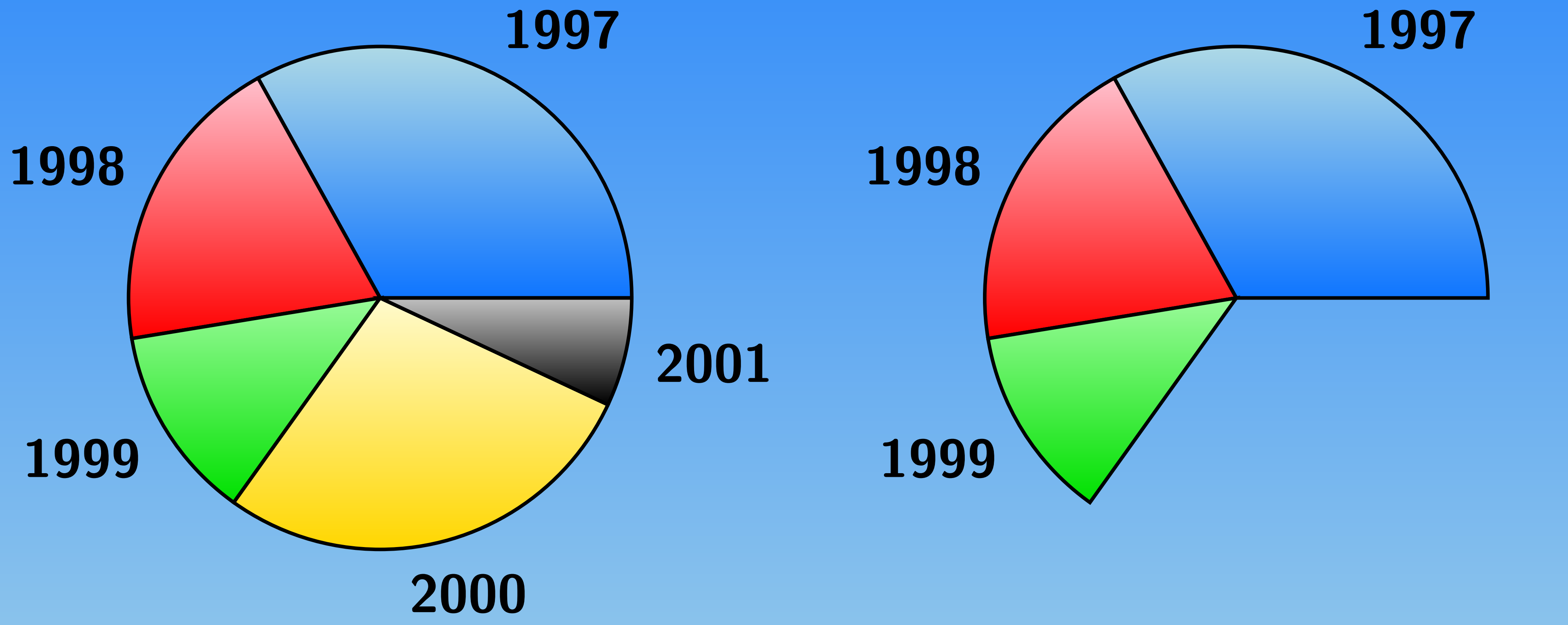


Figure 1: Results of the last five years

An example of cumulative overlays

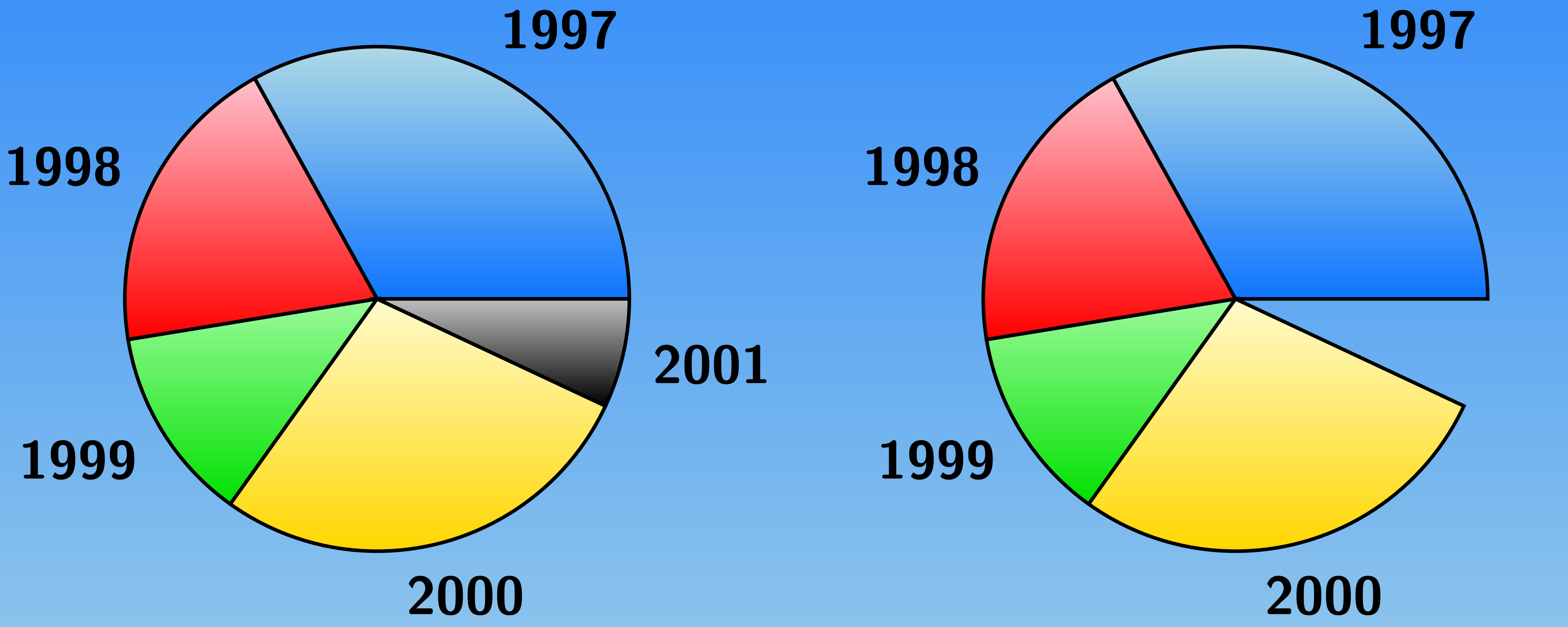
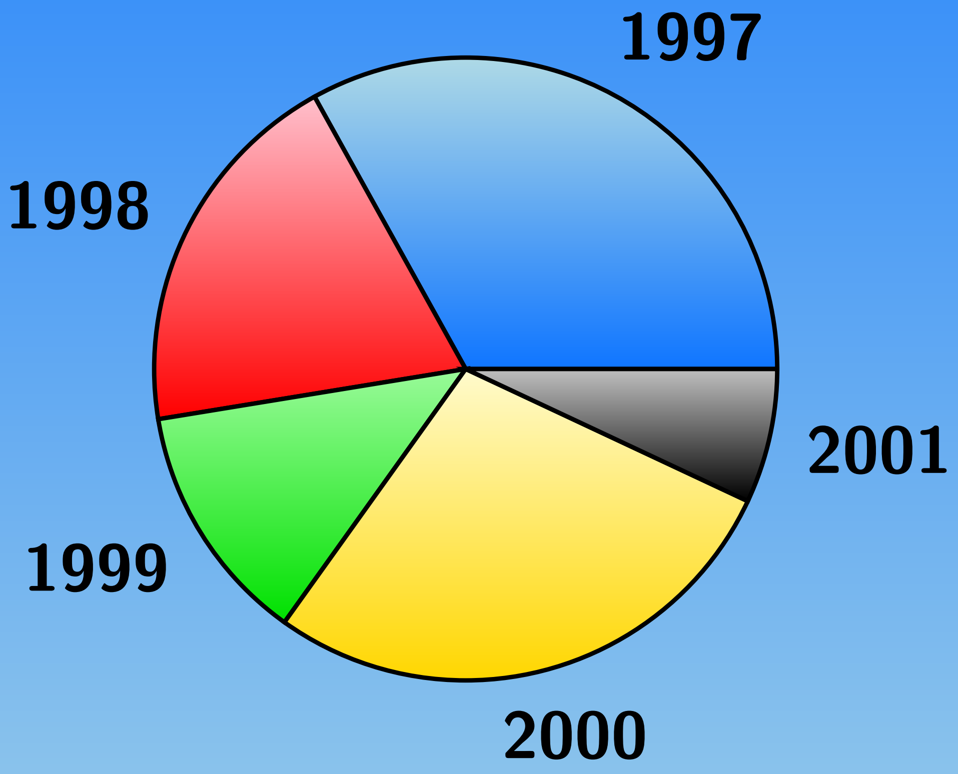


Figure 1: Results of the last five years

An example of cumulative overlays



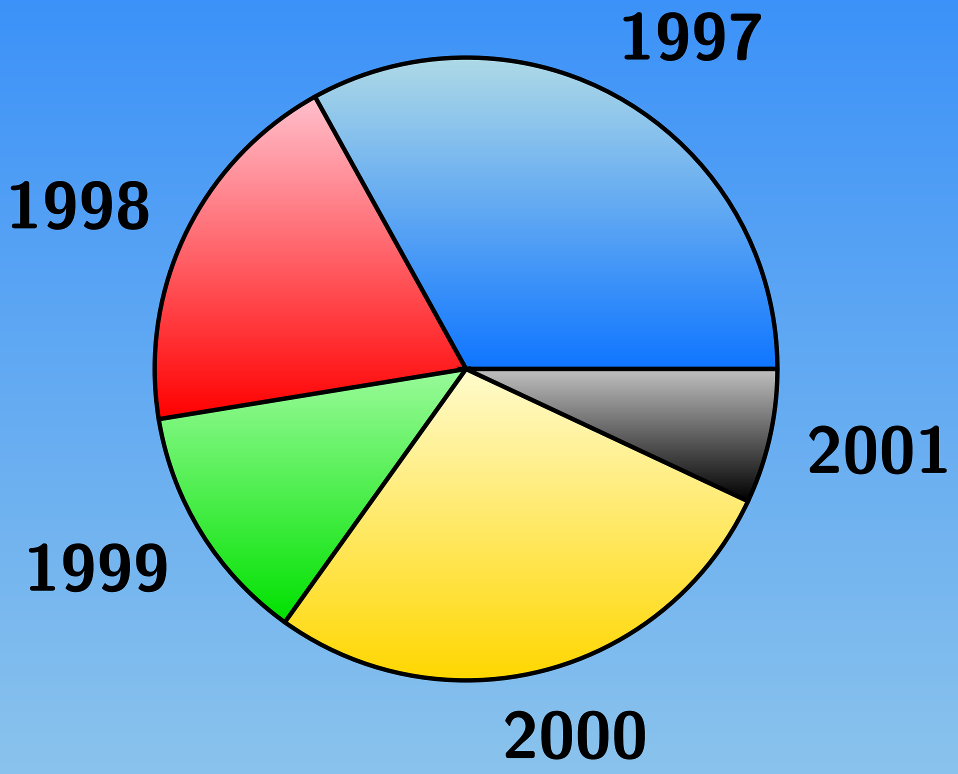


Figure 1: Results of the last five years

An example of cumulative overlays

Figure 2: Main American TV channels

An example of cumulative overlays

Viewership Distribution of the Big Three

Age (years)	NBC	CBS	ABC

Figure 2: Main American TV channels

An example of cumulative overlays

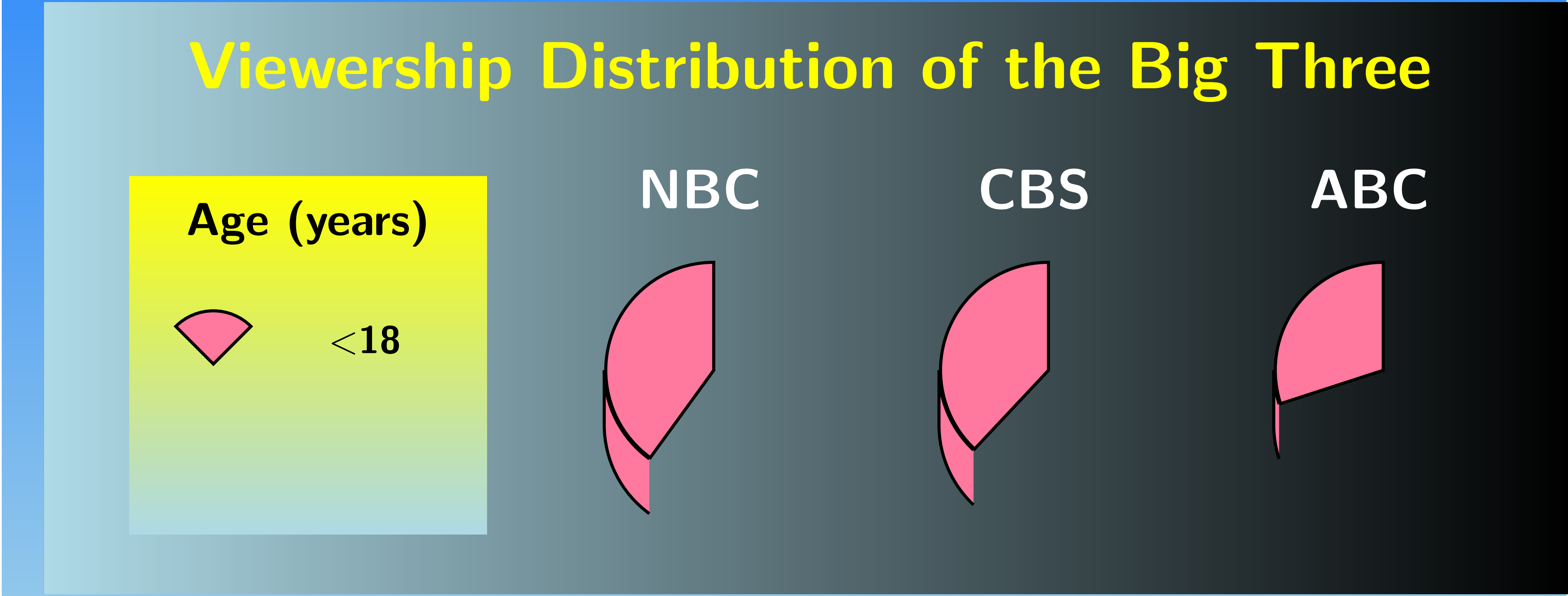


Figure 2: Main American TV channels

An example of cumulative overlays

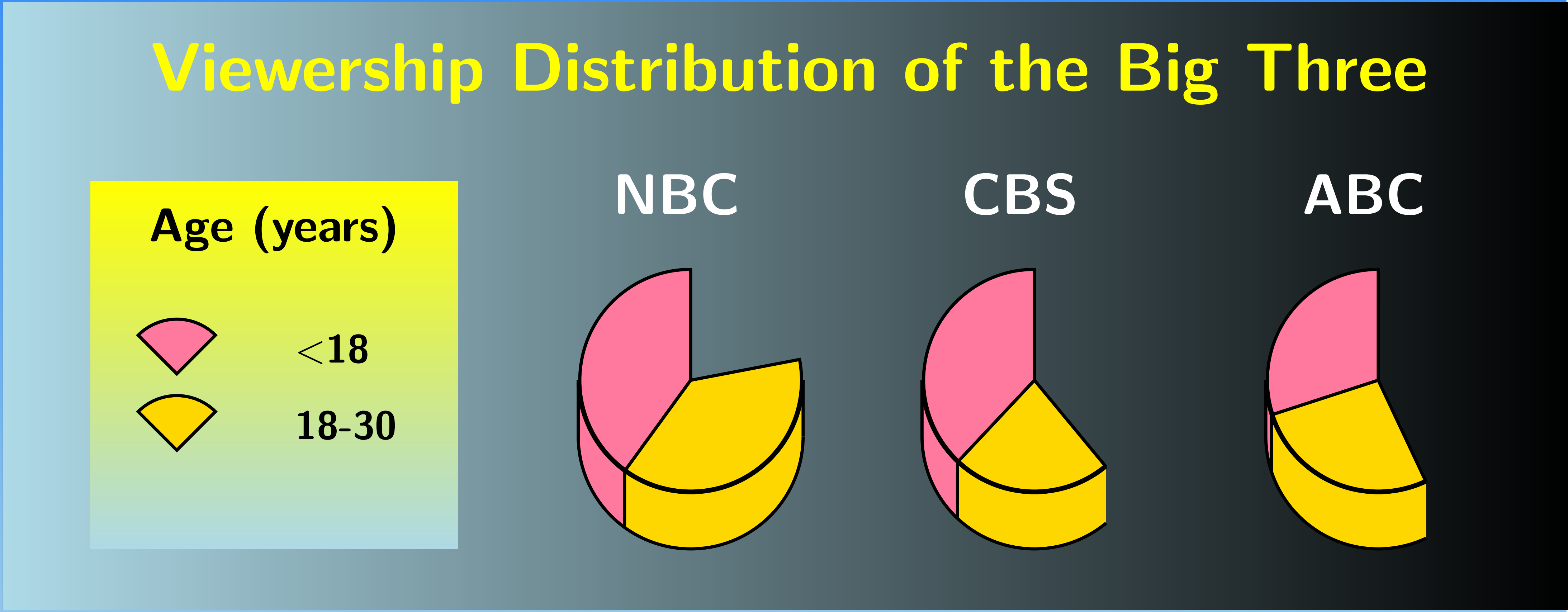


Figure 2: Main American TV channels

An example of cumulative overlays

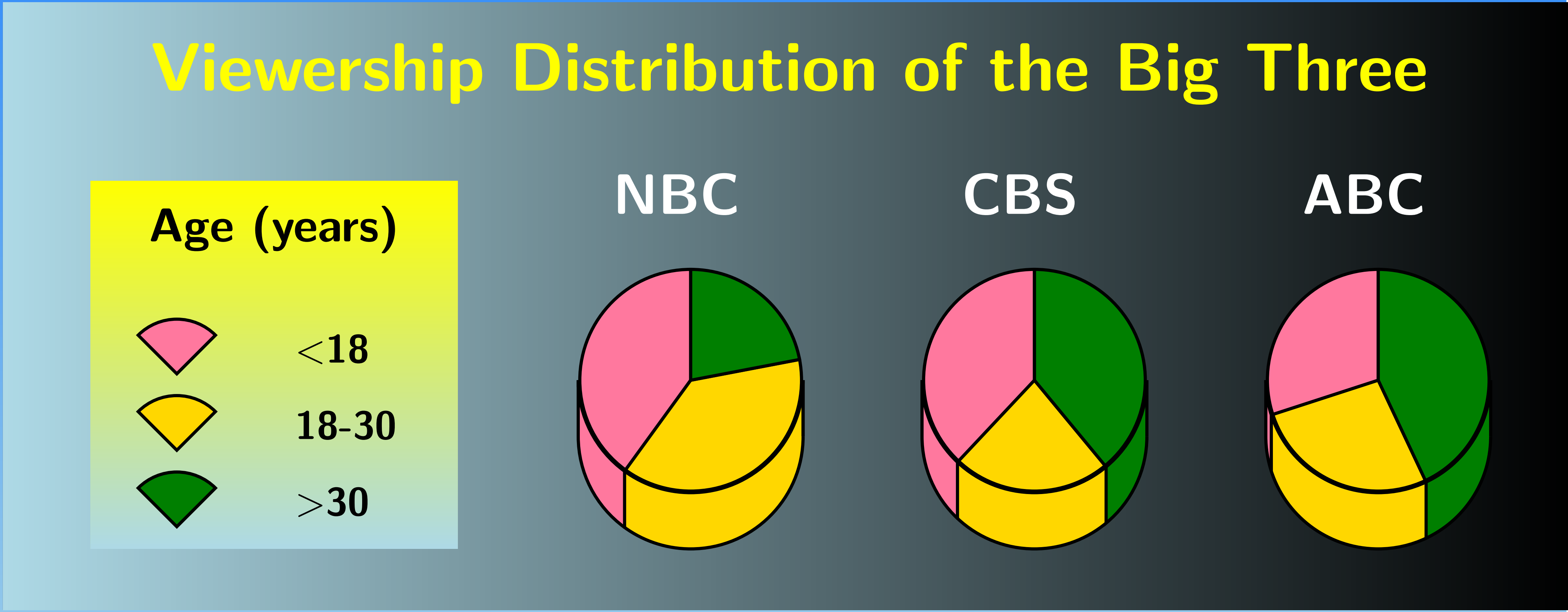


Figure 2: Main American TV channels

An example of cumulative overlays with PSTricks nodes

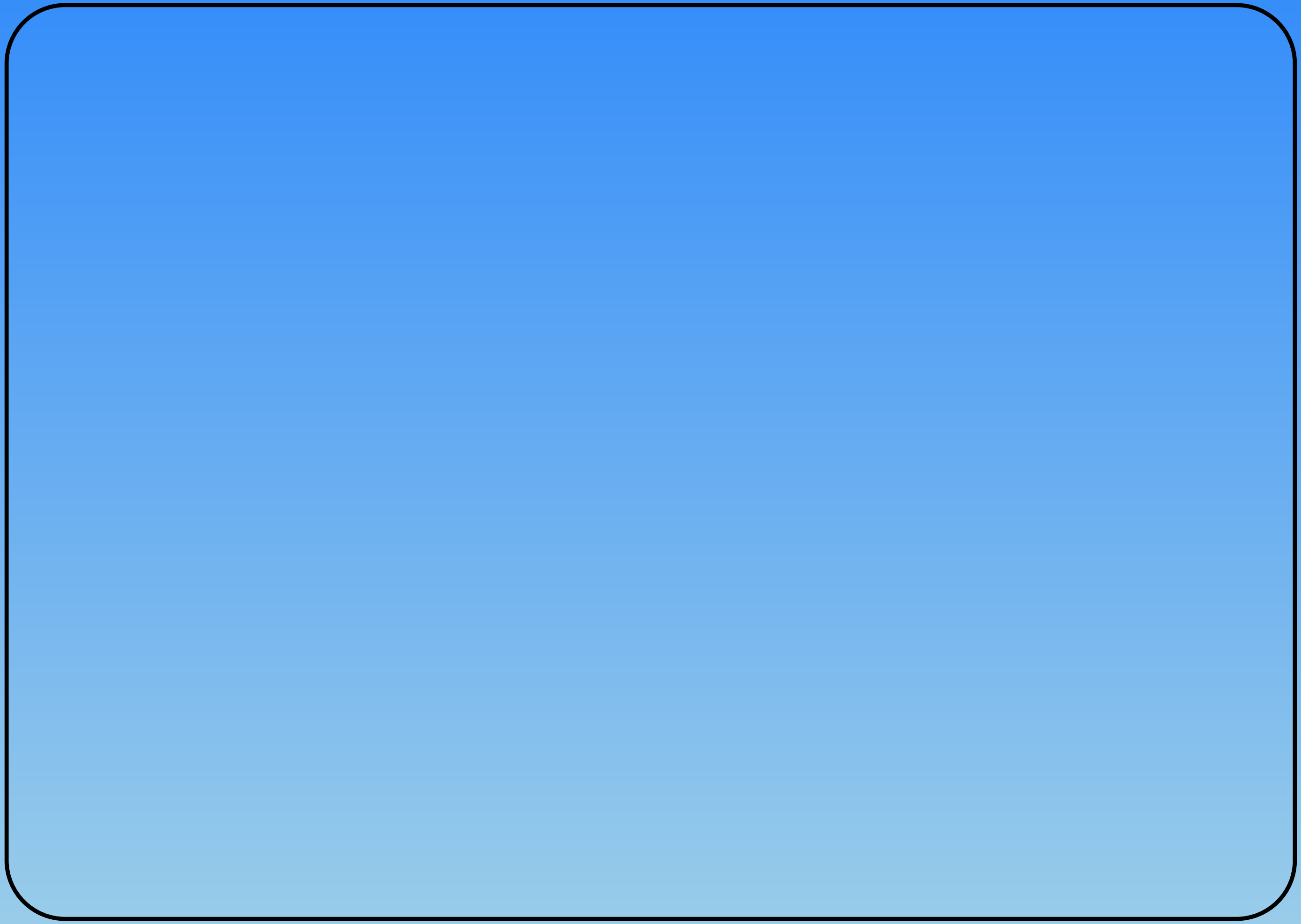


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

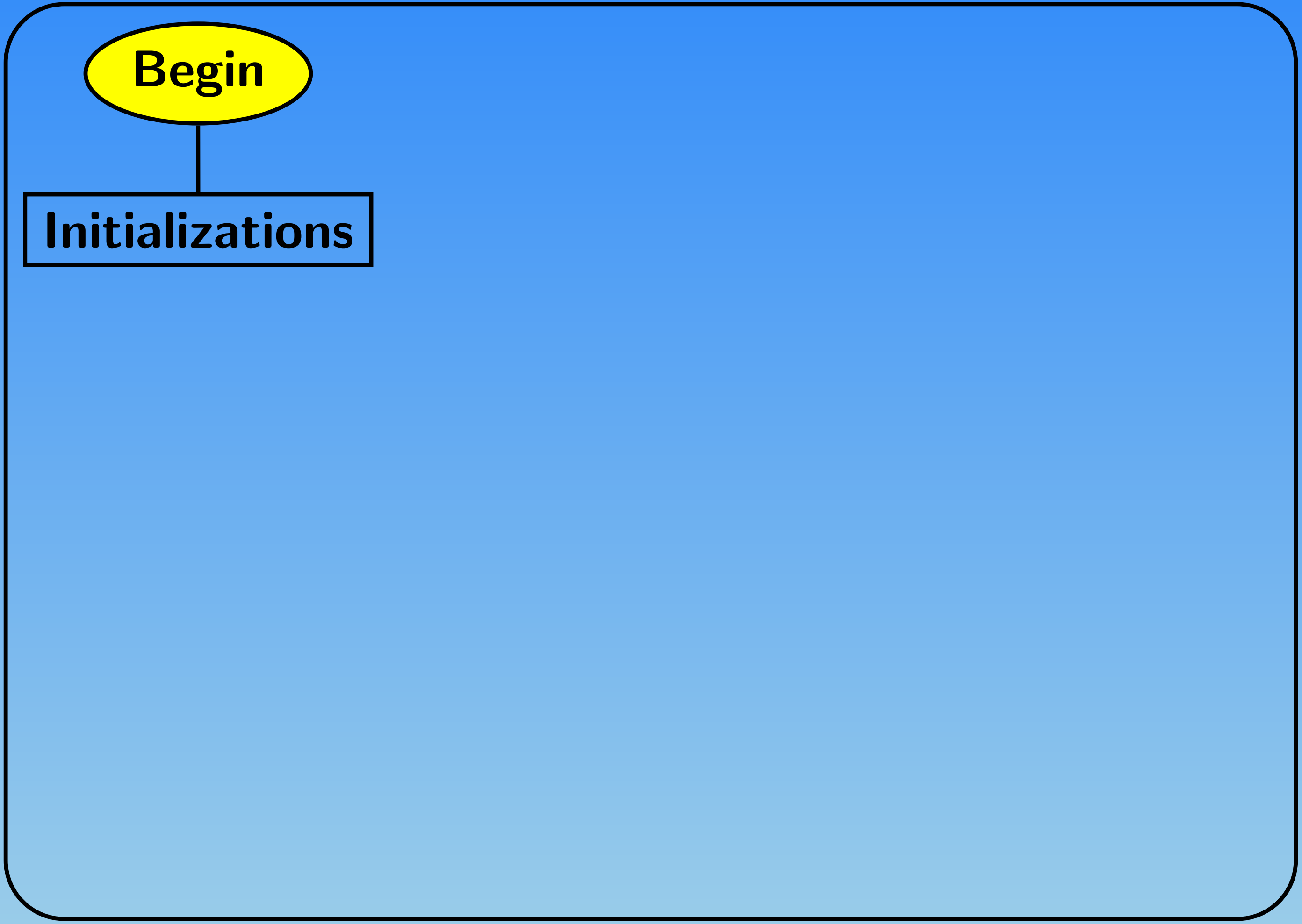


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

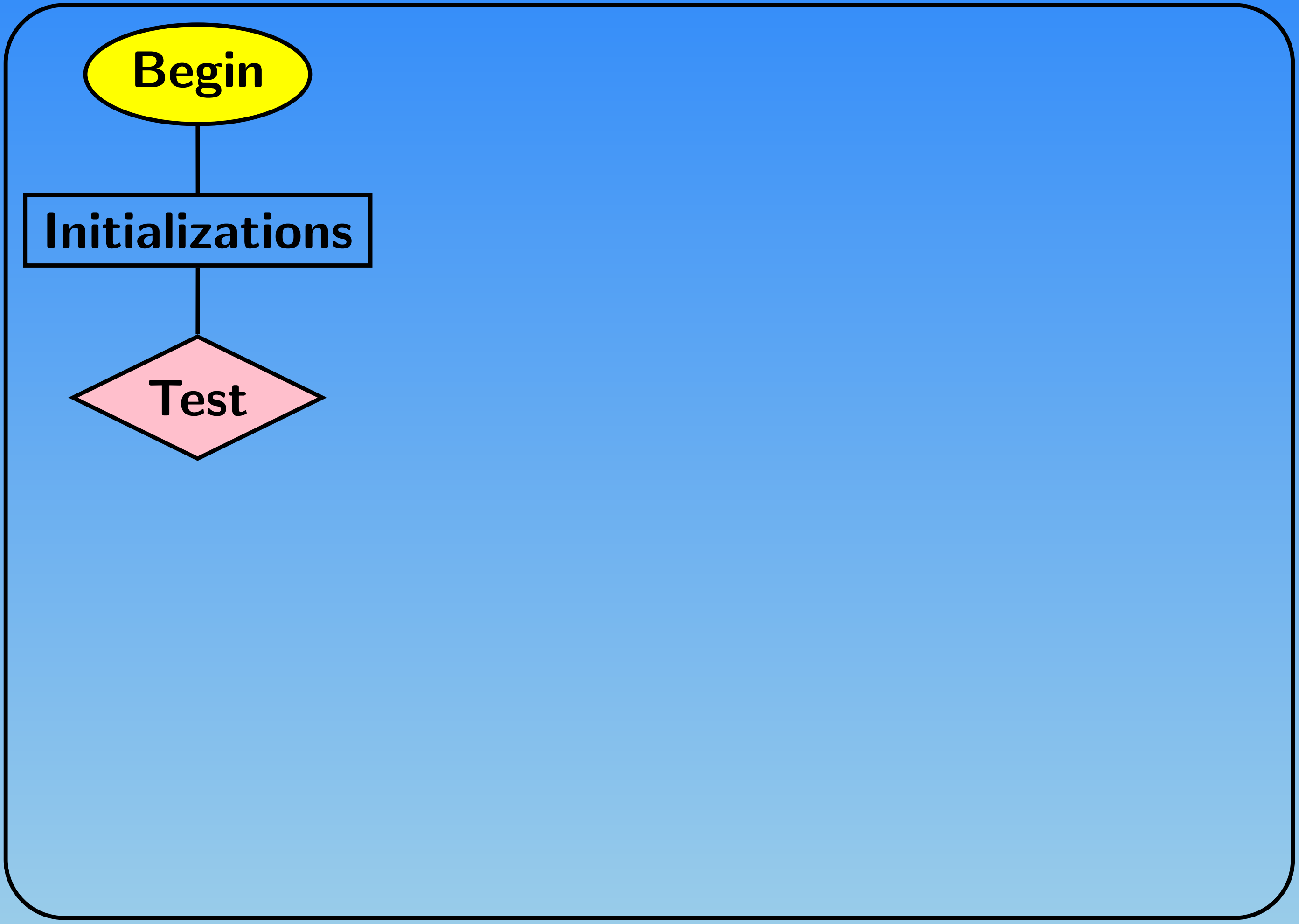


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

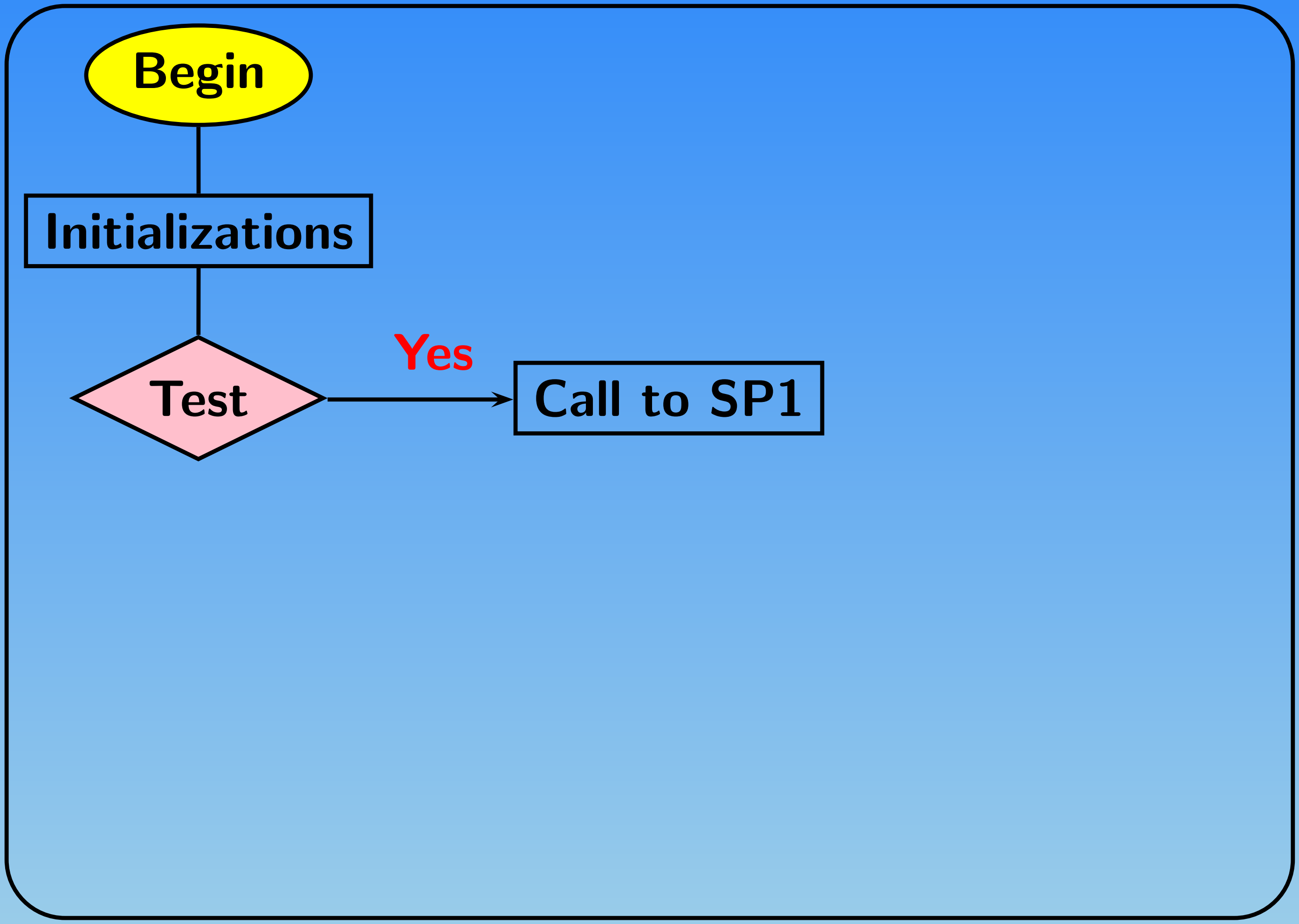


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

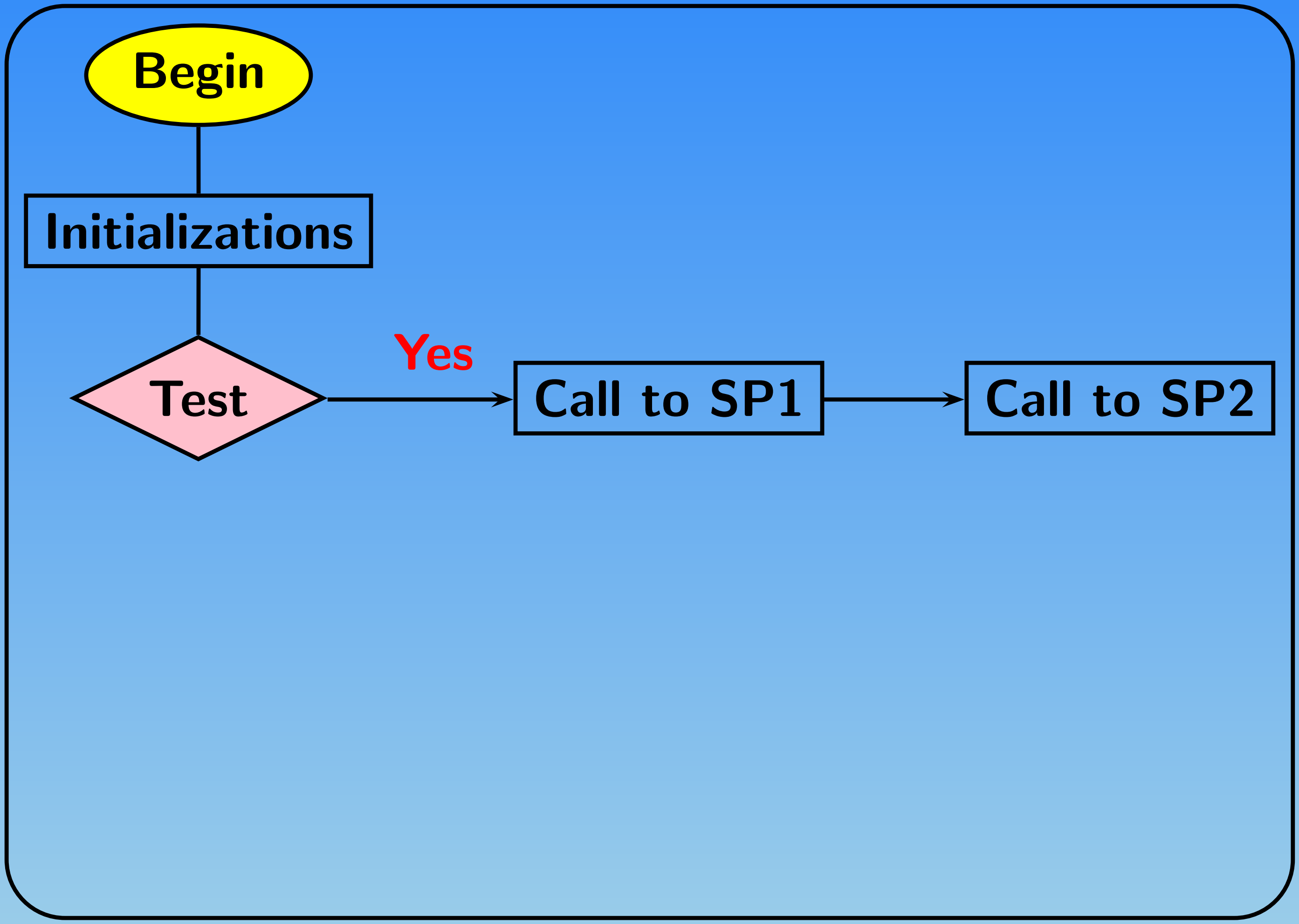


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

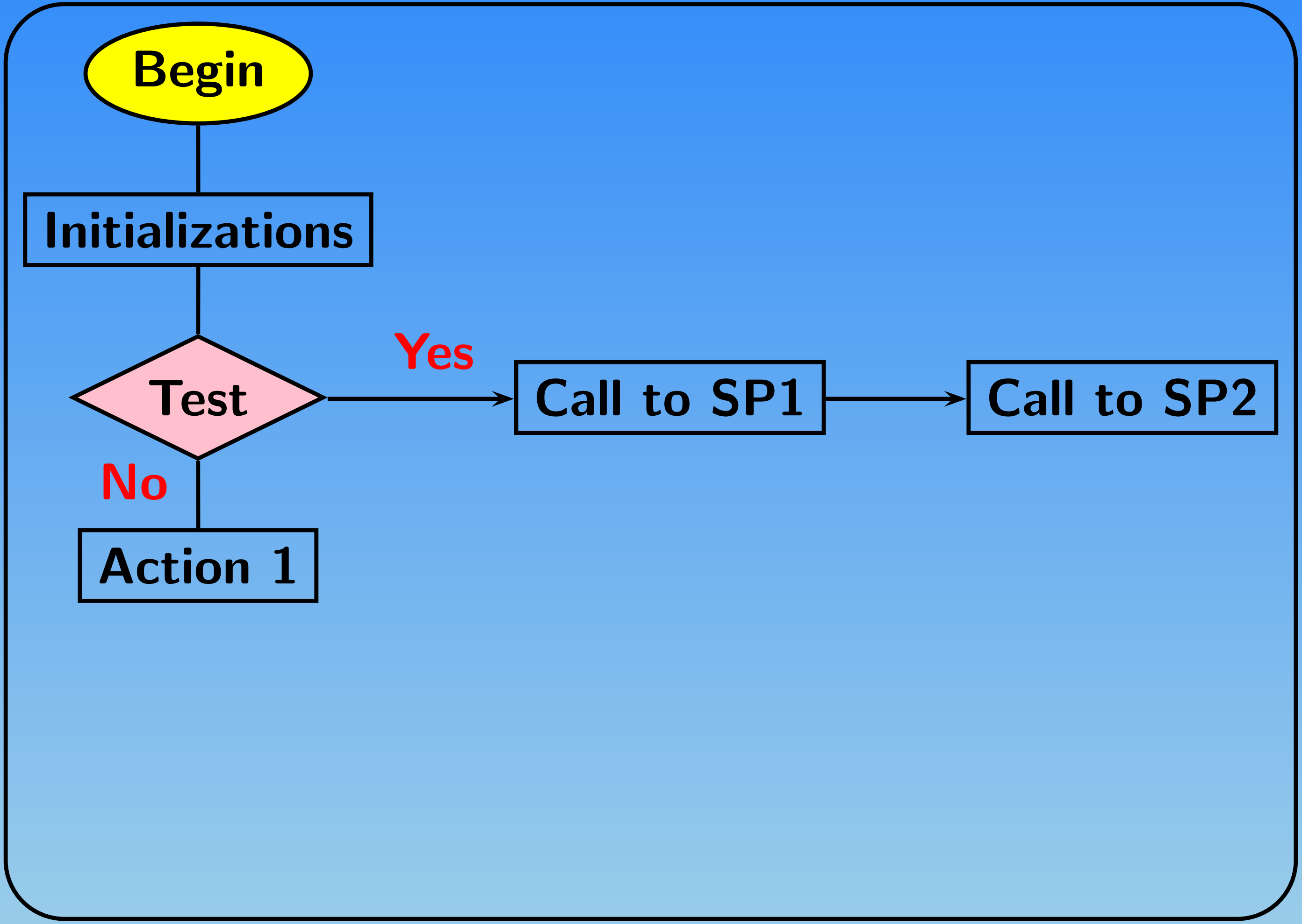


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

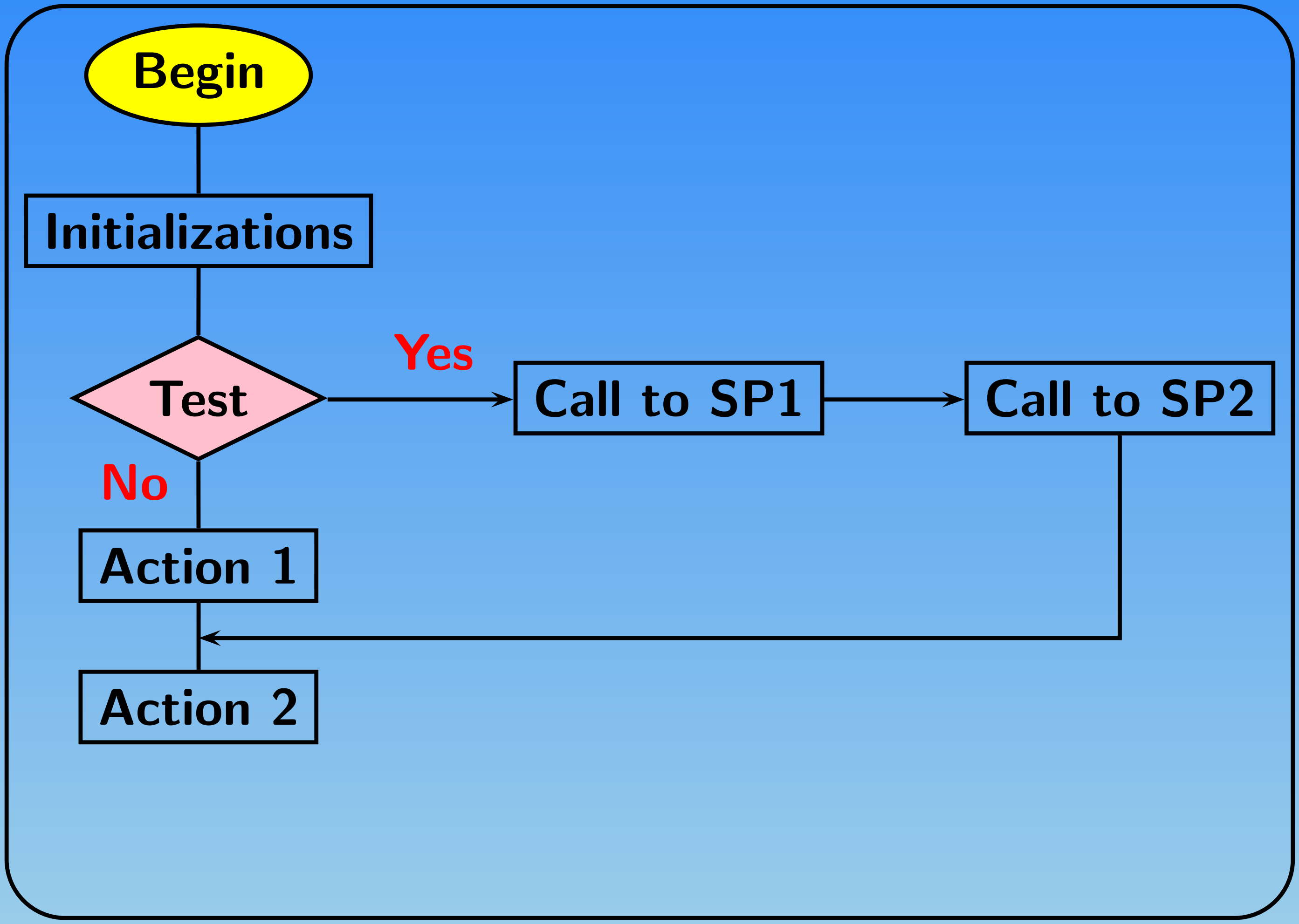


Figure 3: Flow diagram with the `psmatrix` environment

An example of cumulative overlays with PSTricks nodes

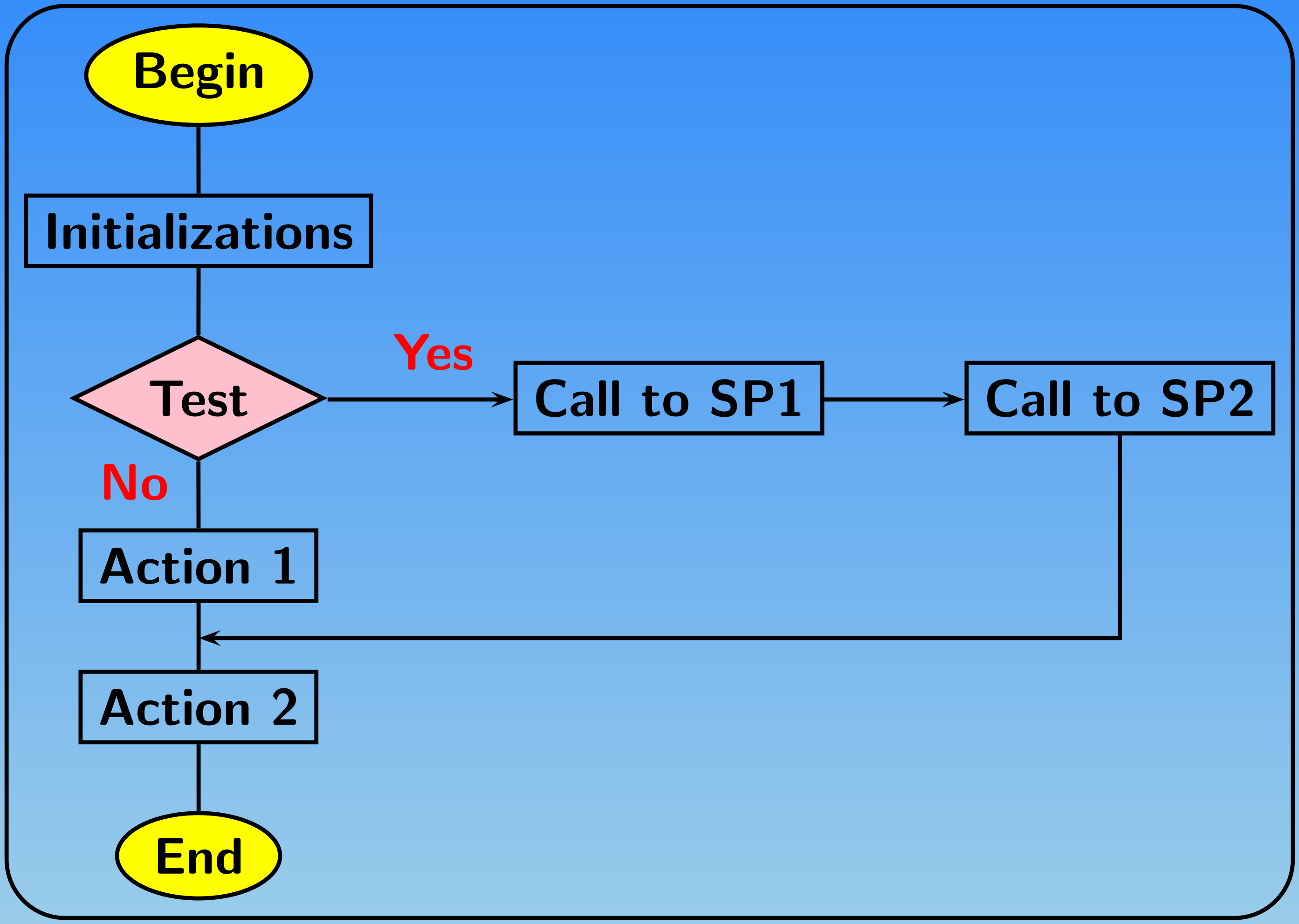


Figure 3: Flow diagram with the `psmatrix` environment

An example of progressive overlays

An example of progressive overlays

- **Miss**

An example of progressive overlays

- ➠ Dear Miss

An example of progressive overlays

- Dear Catherine

An example of progressive overlays

➠ My dear Catherine

An example of progressive overlays

- Darling

An example of progressive overlays

- ➠ My sweet love

An example of progressive overlays

- ➠ My small rabbit

An example of progressive overlays

➠ **My angry rabbit**

An example of progressive overlays

- ➠ My naughty friend

An example of progressive overlays

➠ **Farewell!**

An example of progressive overlays with PSTricks nodes

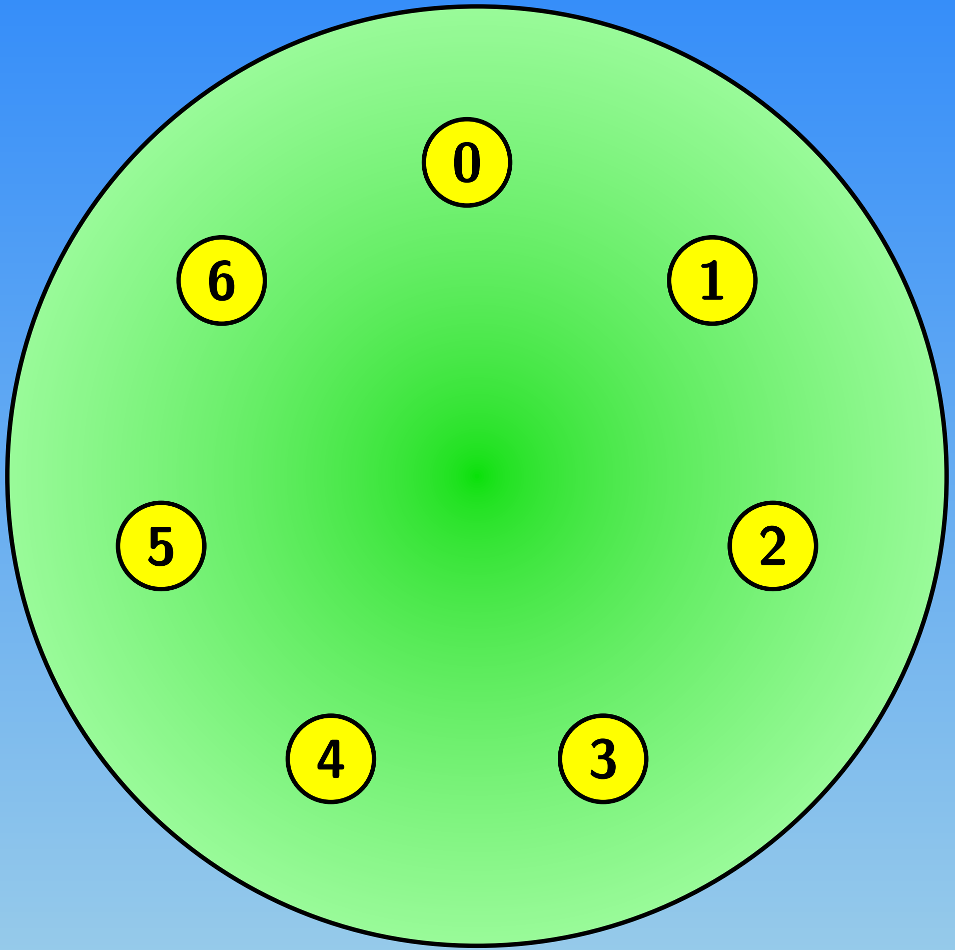


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

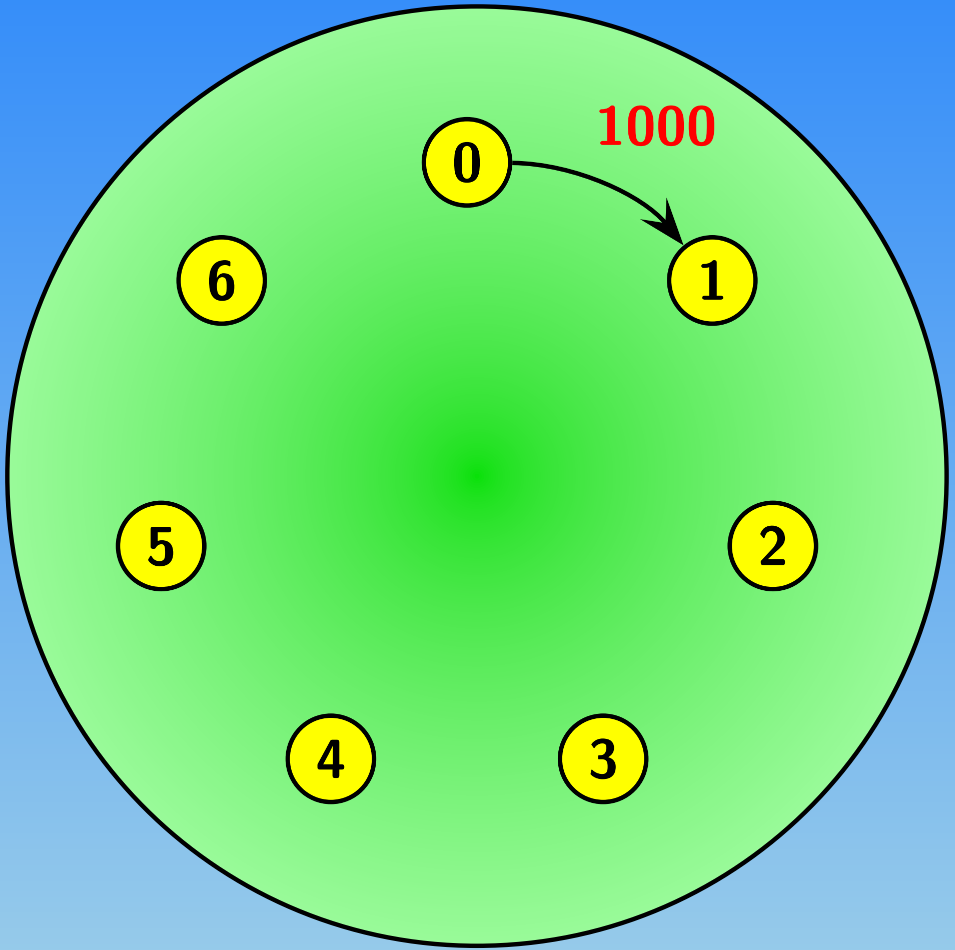


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

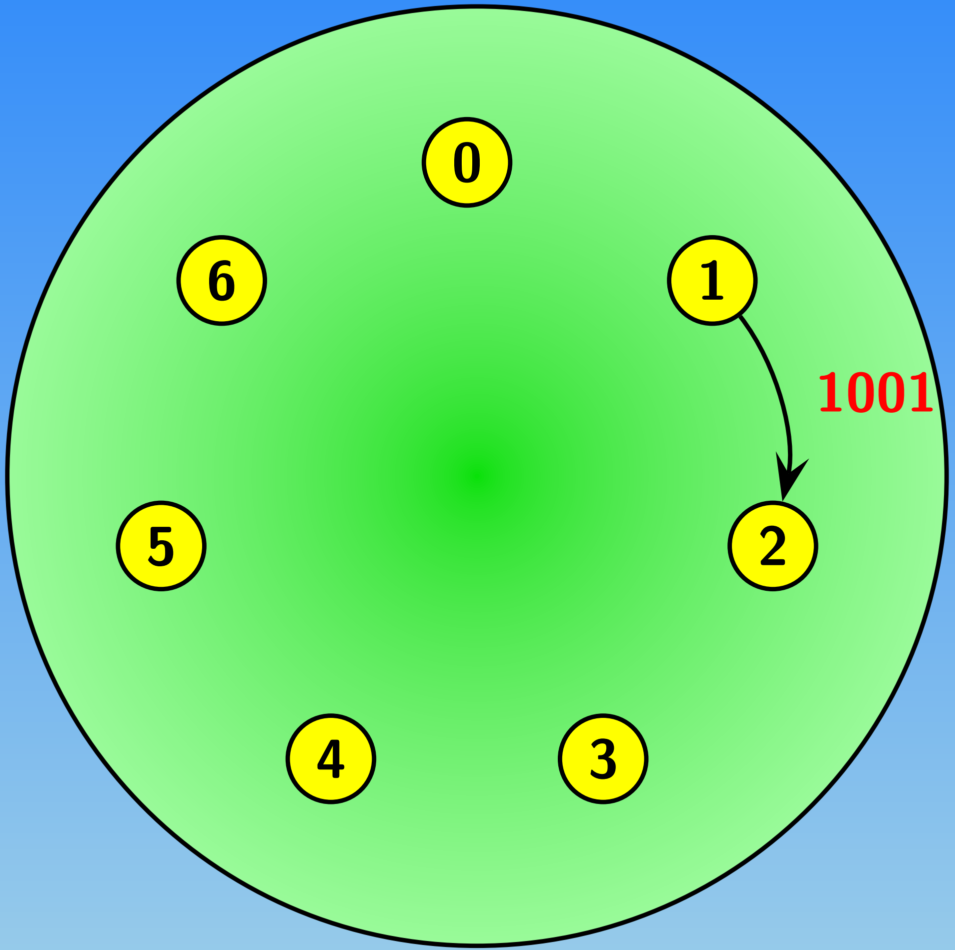


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

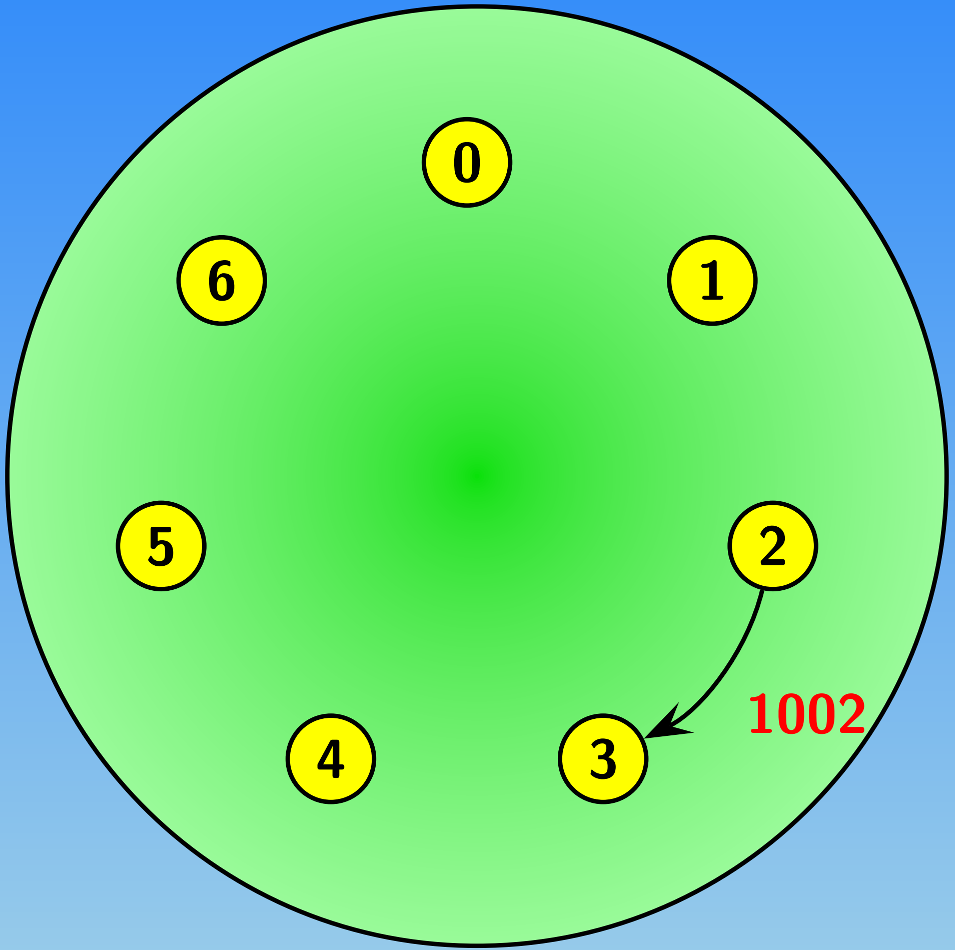


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

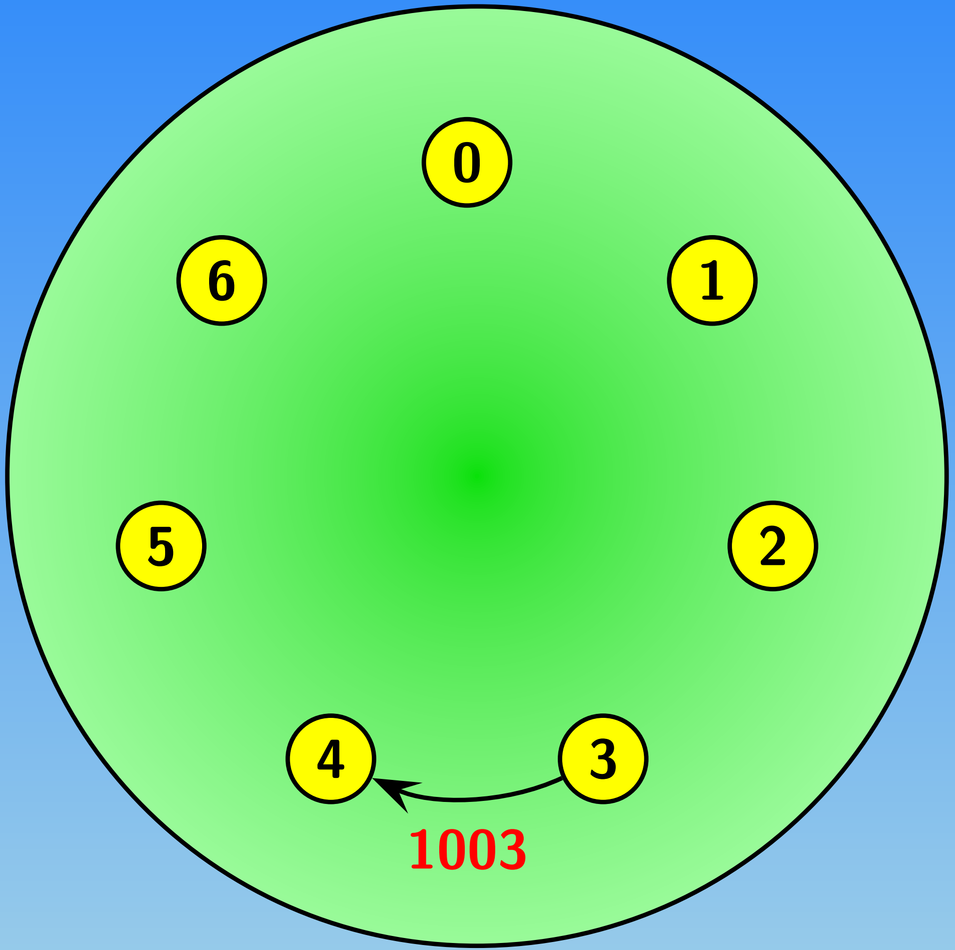


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

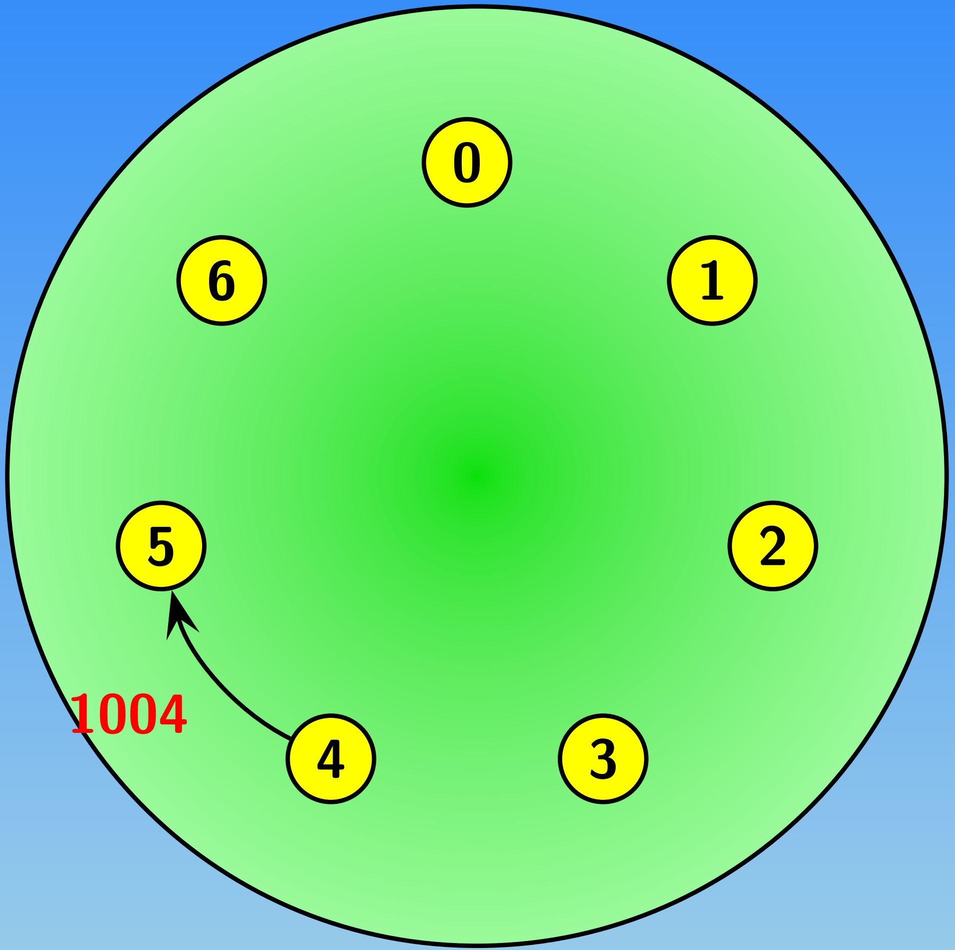


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

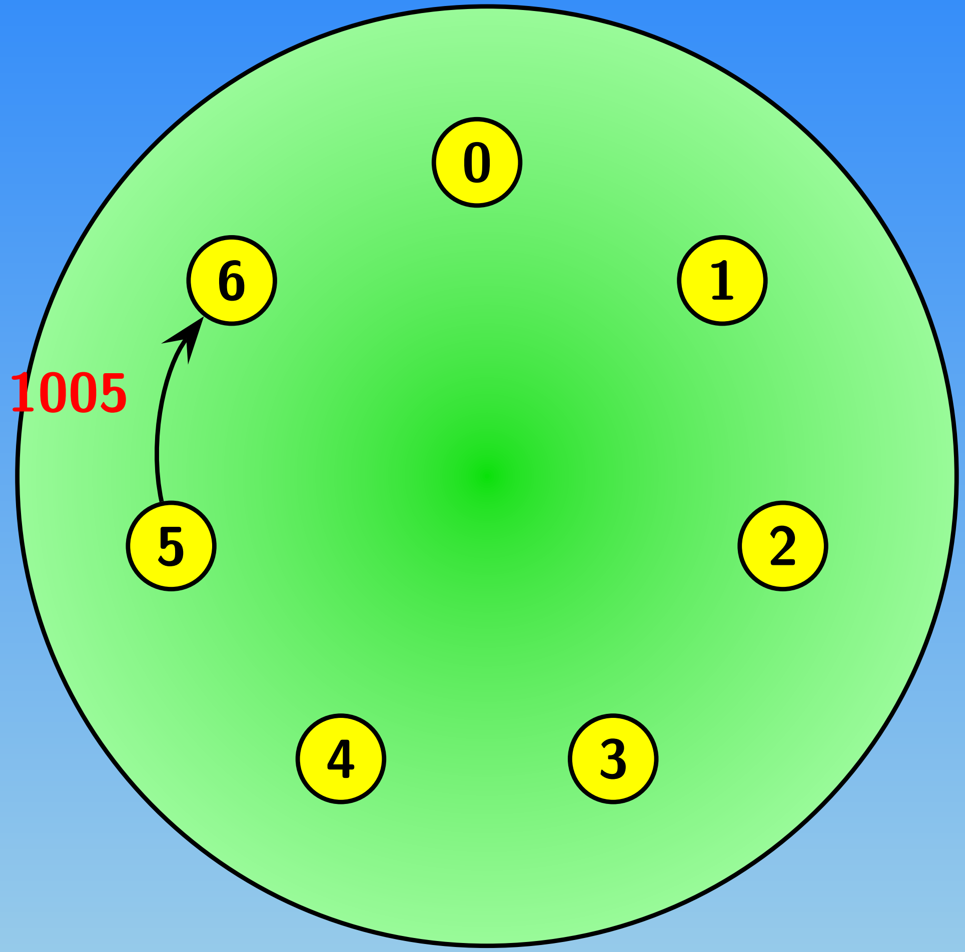


Figure 4: Communication ring

An example of progressive overlays with PSTricks nodes

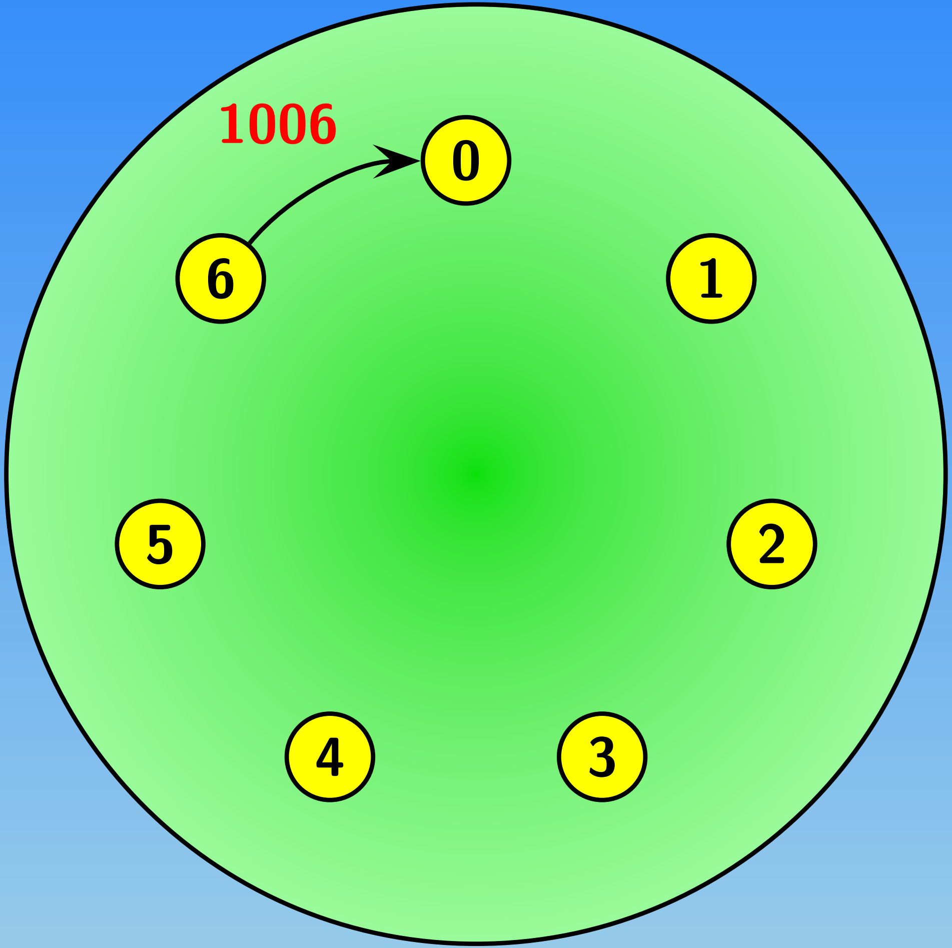


Figure 4: Communication ring

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler

$$\Pi$$

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler

$$\Pi = \overset{\boxed{= 2.44949}}{\sqrt{6}}$$

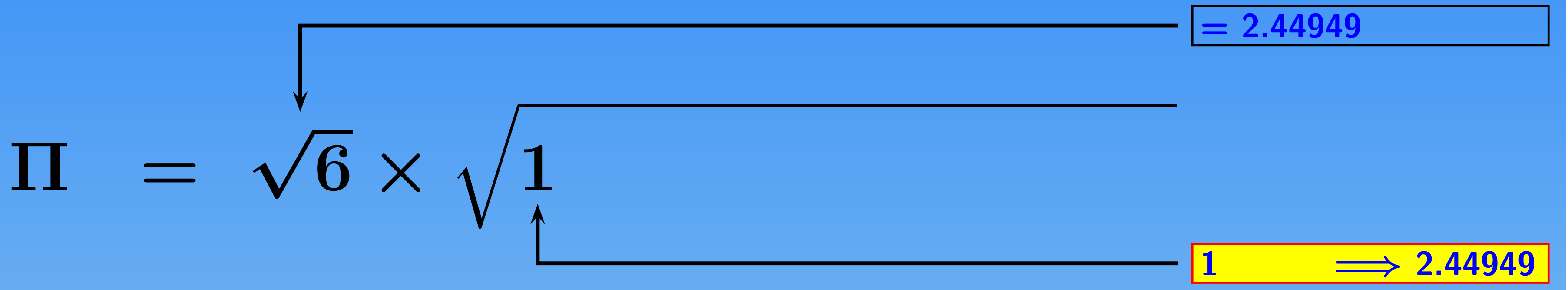
An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler

$$\Pi \;=\; \overset{\boxed{= 2.44949}}{\sqrt{6}} \times \sqrt{\qquad\qquad\qquad\qquad}$$

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler



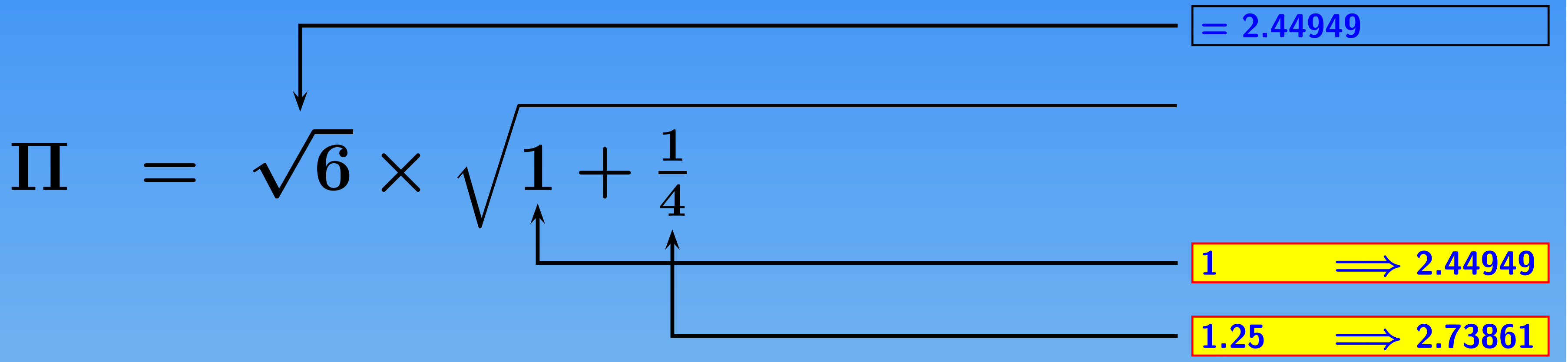
$$\Pi = \sqrt{6} \times \sqrt{1}$$

$= 2.44949$

$1 \Longrightarrow 2.44949$

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler



$$\Pi = \sqrt{6} \times \sqrt{1 + \frac{1}{4}}$$

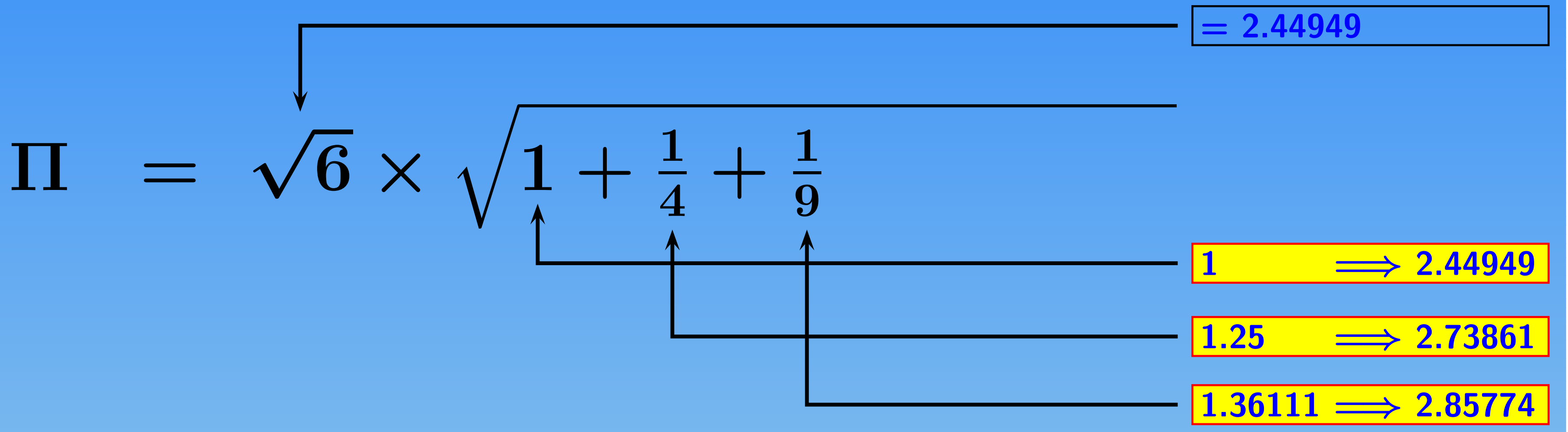
$= 2.44949$

1 $\Longrightarrow$ 2.44949

1.25 $\Longrightarrow$ 2.73861

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler



$$\Pi = \sqrt{6} \times \sqrt{1 + \frac{1}{4} + \frac{1}{9}}$$

$= 2.44949$

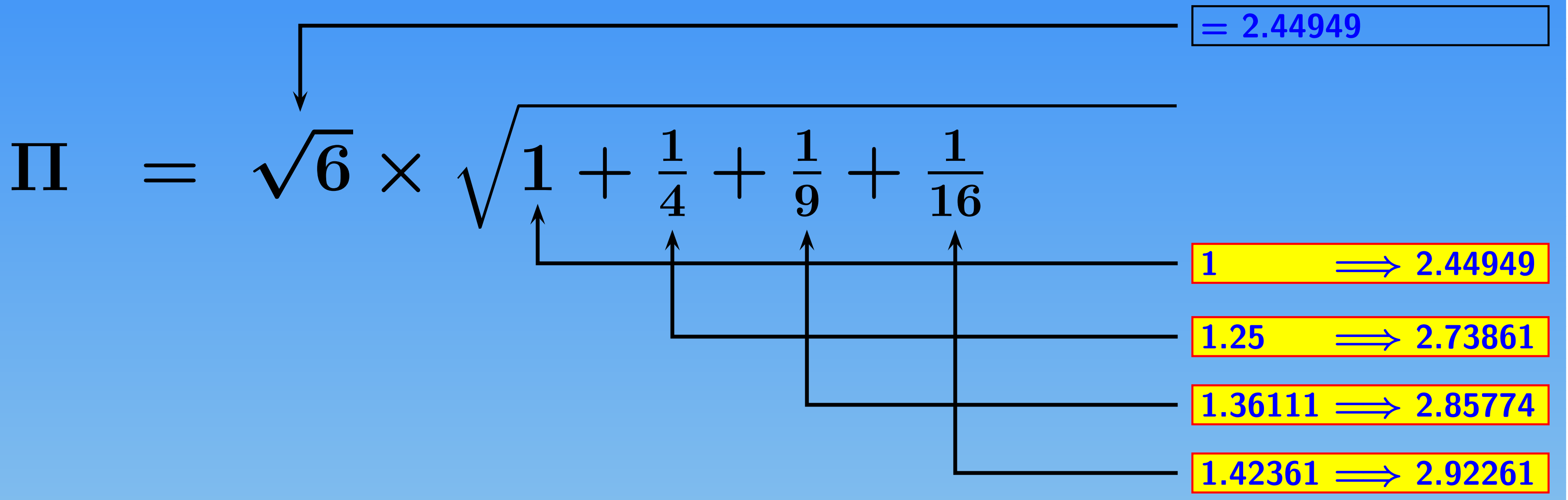
$1 \Longrightarrow 2.44949$

$1.25 \Longrightarrow 2.73861$

$1.36111 \Longrightarrow 2.85774$

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler



$$\Pi = \sqrt{6} \times \sqrt{1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16}}$$

$= 2.44949$

$1 \Longrightarrow 2.44949$

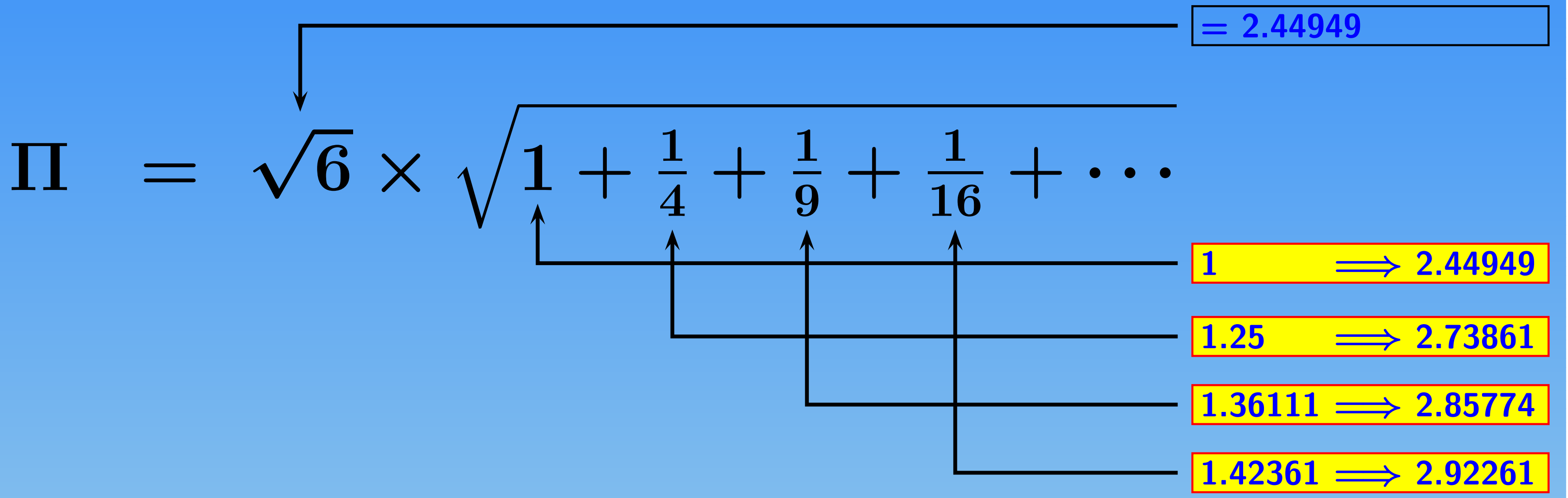
$1.25 \Longrightarrow 2.73861$

$1.36111 \Longrightarrow 2.85774$

$1.42361 \Longrightarrow 2.92261$

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler



$$\Pi = \sqrt{6} \times \sqrt{1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16} + \cdots}$$

$= 2.44949$

$1 \Longrightarrow 2.44949$

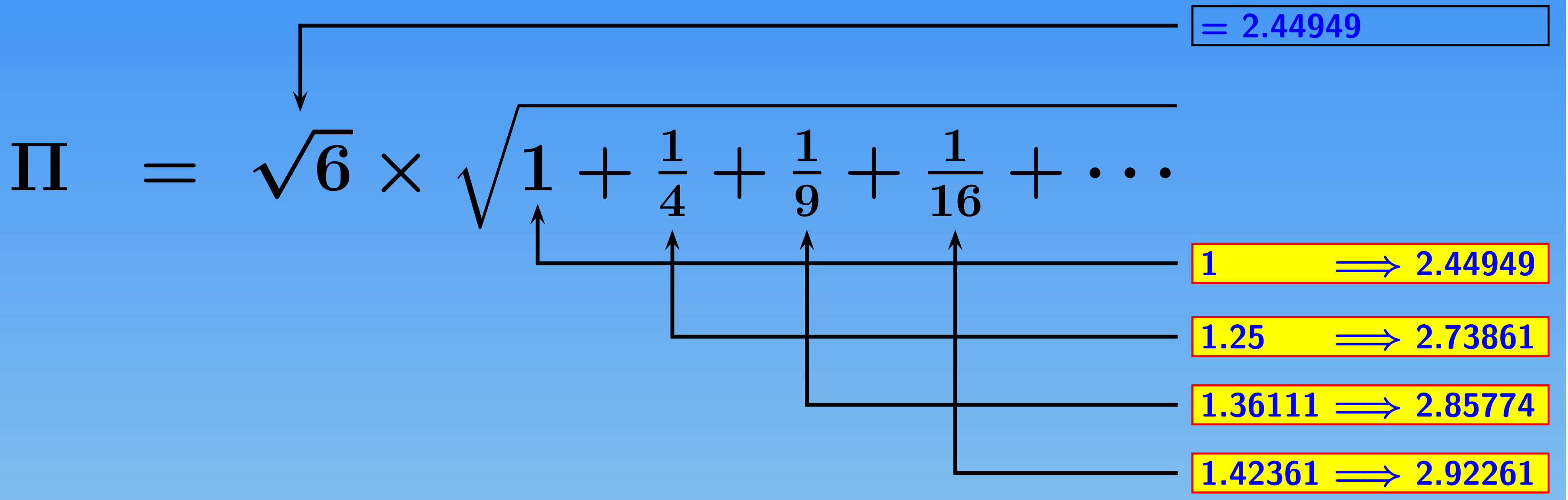
$1.25 \Longrightarrow 2.73861$

$1.36111 \Longrightarrow 2.85774$

$1.42361 \Longrightarrow 2.92261$

An example of equations with (cumulative) annotations

A formula for Π from Leonhard Euler



$$\Pi = \sqrt{6} \times \sqrt{1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16} + \cdots}$$

$= 2.44949$

$1 \Longrightarrow 2.44949$

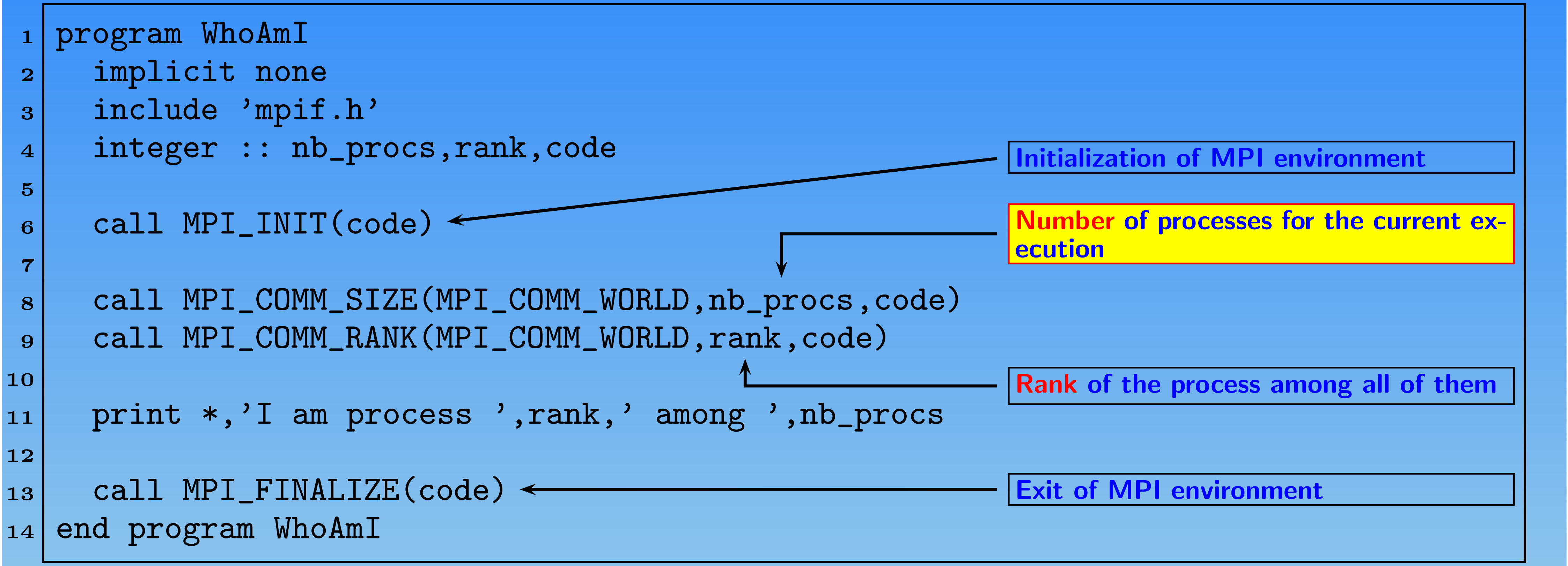
$1.25 \Longrightarrow 2.73861$

$1.36111 \Longrightarrow 2.85774$

$1.42361 \Longrightarrow 2.92261$

$$= \left(6 \sum_{n=1}^{\infty} \frac{1}{n^2}\right)^{\frac{1}{2}}$$

An example of listings with annotations



```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,'I am process ',rank,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

Number of processes for the current execution

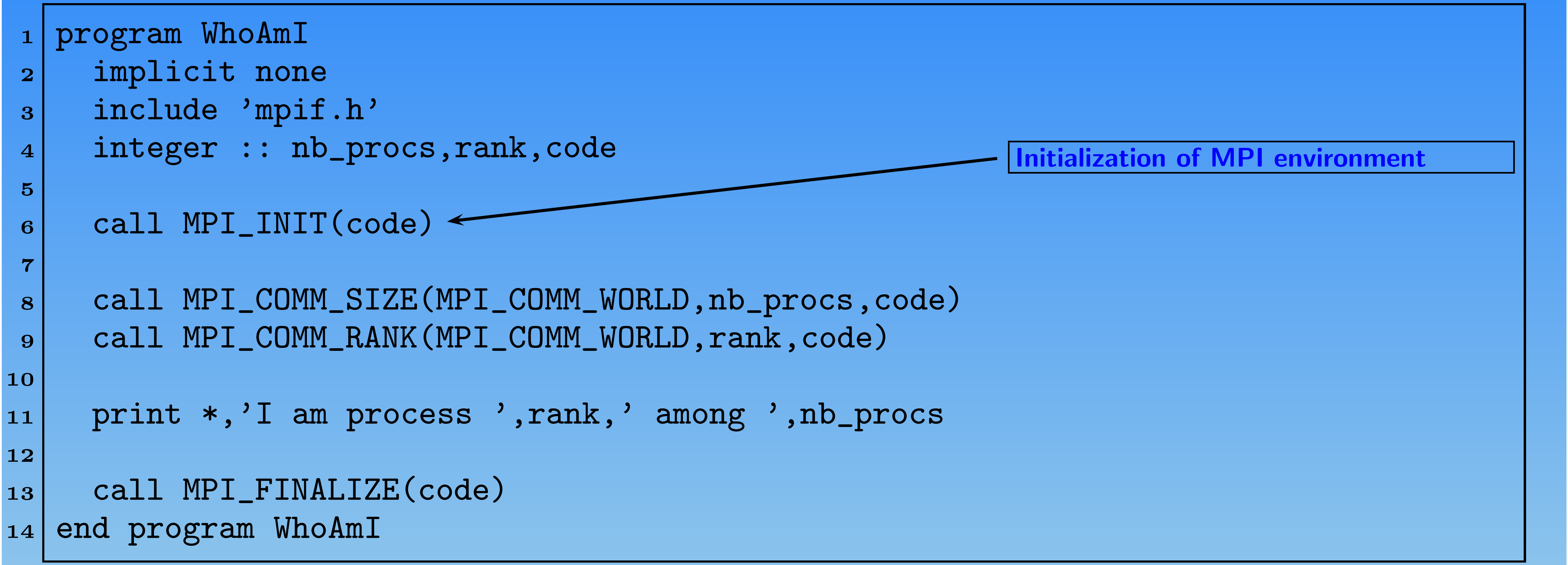
Rank of the process among all of them

Exit of MPI environment

More examples of listings with annotations

```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,'I am process ',rank,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

More examples of listings with annotations



```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,'I am process ',rank,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

More examples of listings with annotations

```
program WhoAmI
  implicit none
  include ’mpif.h’
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,’I am process ’,rank,’ among ’,nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Number of processes for the current execution

More examples of listings with annotations

```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,'I am process ',rank,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Rank of the process among all of them

More examples of listings with annotations

```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,'I am process ',rank,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Exit of MPI environment

```
program WhoAmI
  implicit none
  include ’mpif.h’
  integer :: nb_procs,rank,code







  print *,’I am process ’,    ,’ among ’,


end program WhoAmI
```

```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)




  print *,'I am process ',    ,' among ',


end program WhoAmI
```

```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)




  print *,'I am process ',     ,' among ',


end program WhoAmI
```

Initialization of MPI environment

```
program WhoAmI
  implicit none
  include ’mpif.h’
  integer :: nb_procs,rank,code

  call MPI_INIT(code)




  print *,’I am process ’,     ,’ among ’,

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

```
program WhoAmI
  implicit none
  include ’mpif.h’
  integer :: nb_procs,rank,code

  call MPI_INIT(code)




  print *,’I am process ’,     ,’ among ’,

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

Exit of MPI environment

```
program WhoAmI
  implicit none
  include ’mpif.h’
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)


  print *,’I am process ’,     ,’ among ’,nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

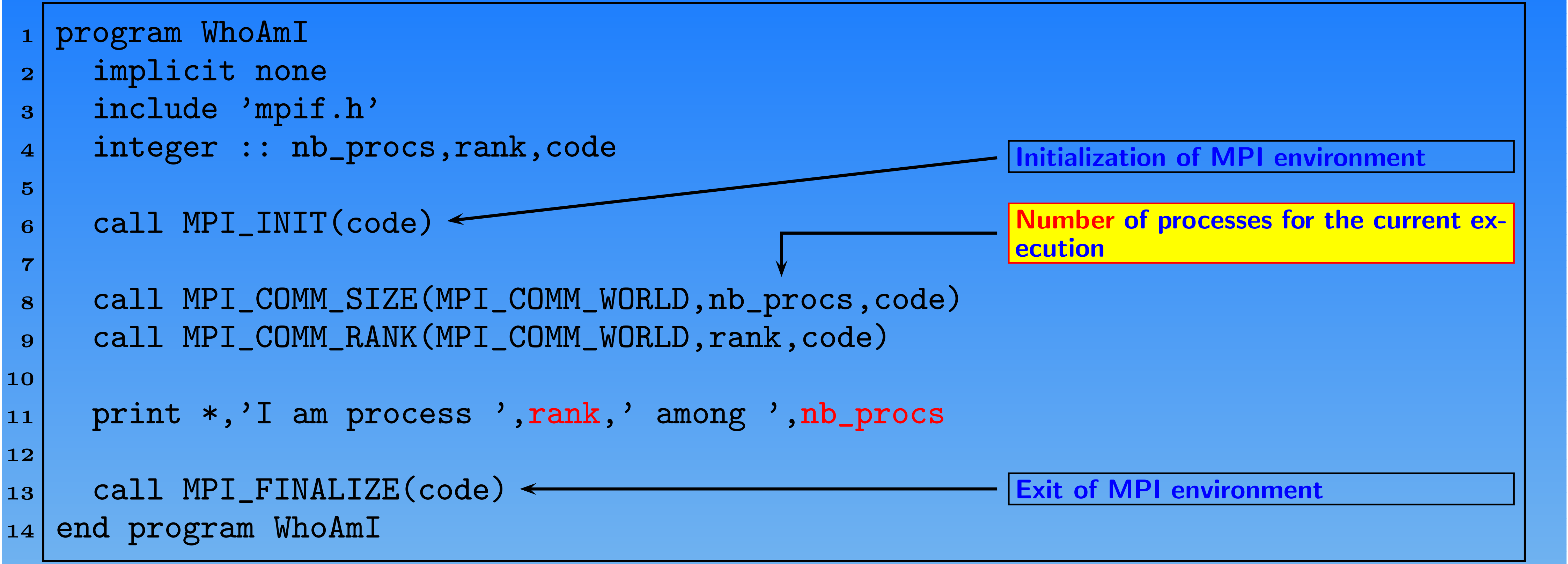
Exit of MPI environment

```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)


  print *,'I am process ',    ,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

Number of processes for the current execution

Exit of MPI environment

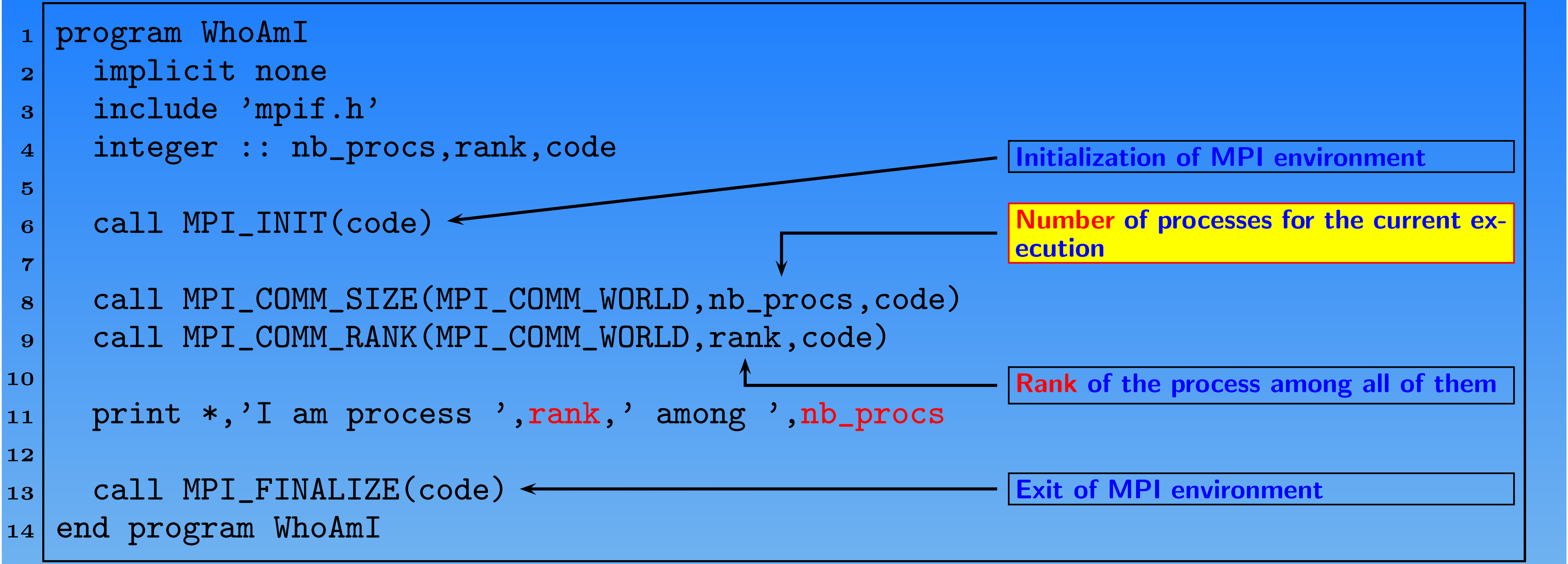


```
program WhoAmI
  implicit none
  include ’mpif.h’
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,’I am process ’,rank,’ among ’,nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

Number of processes for the current execution

Exit of MPI environment



```
program WhoAmI
  implicit none
  include 'mpif.h'
  integer :: nb_procs,rank,code

  call MPI_INIT(code)

  call MPI_COMM_SIZE(MPI_COMM_WORLD,nb_procs,code)
  call MPI_COMM_RANK(MPI_COMM_WORLD,rank,code)

  print *,'I am process ',rank,' among ',nb_procs

  call MPI_FINALIZE(code)
end program WhoAmI
```

Initialization of MPI environment

Number of processes for the current execution

Rank of the process among all of them

Exit of MPI environment

Navigation Bars and Panels

Demonstration of Navigation Bars and Panels

Animations

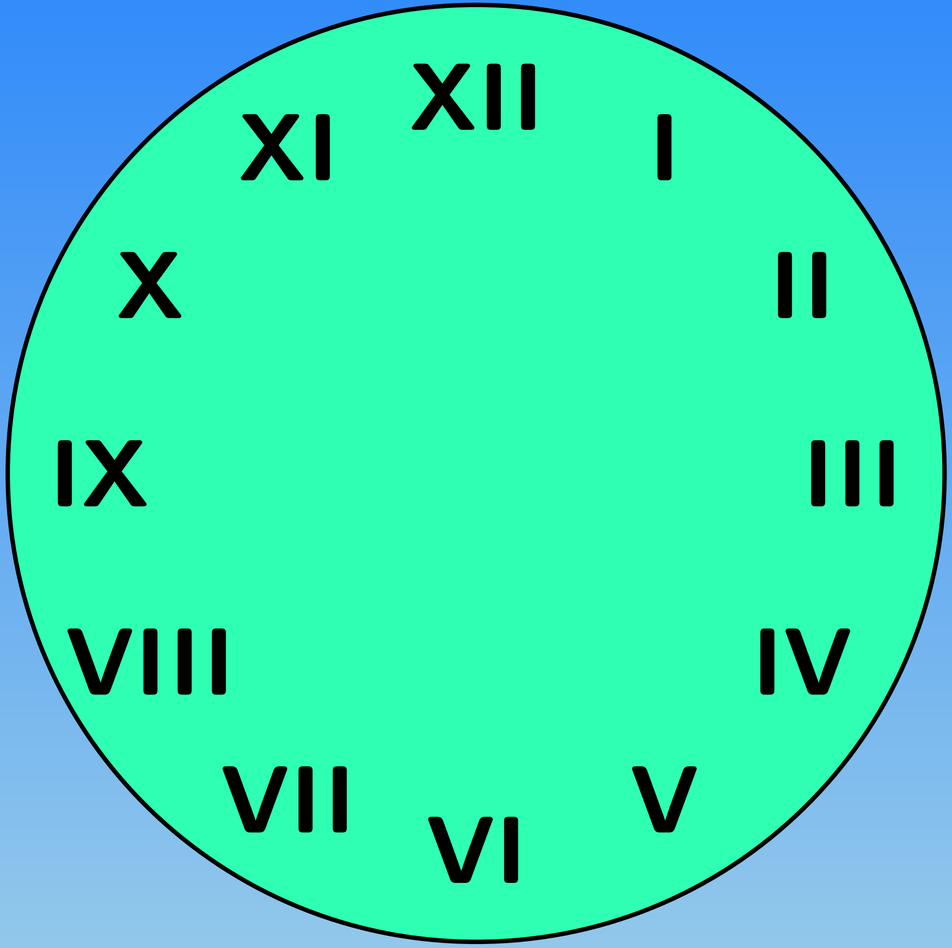


Figure 1: Clock

Animations

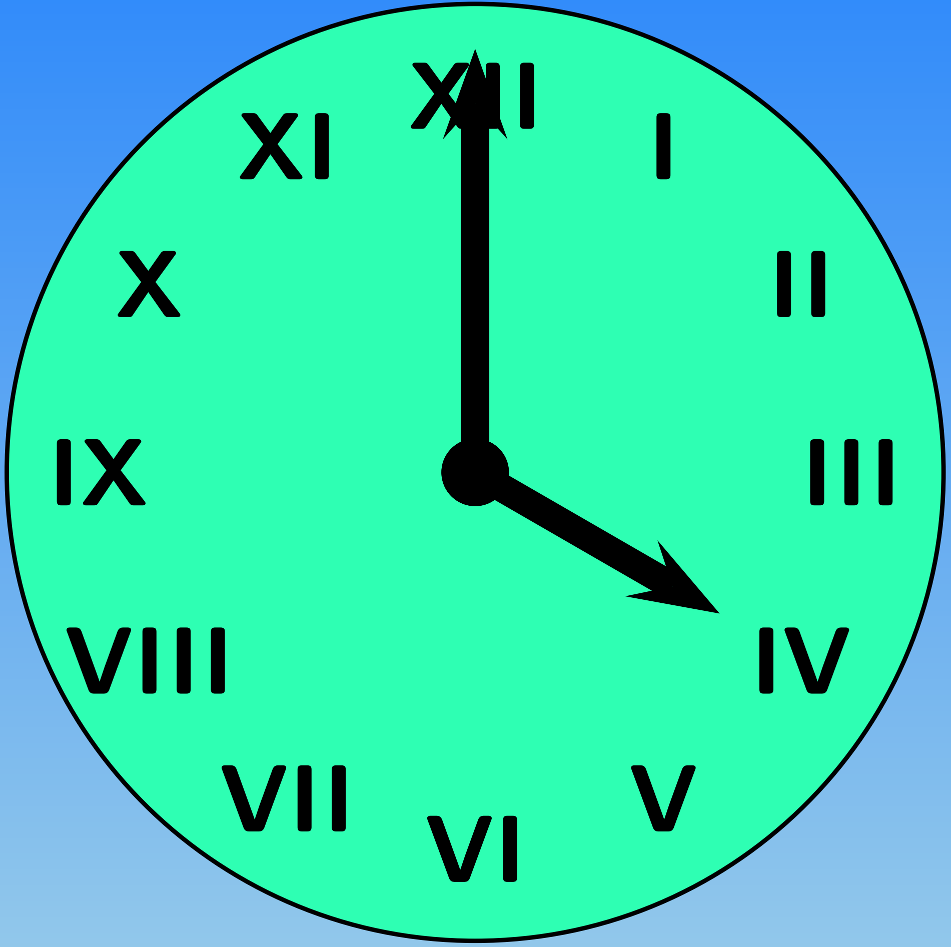


Figure 1: Clock

Animations

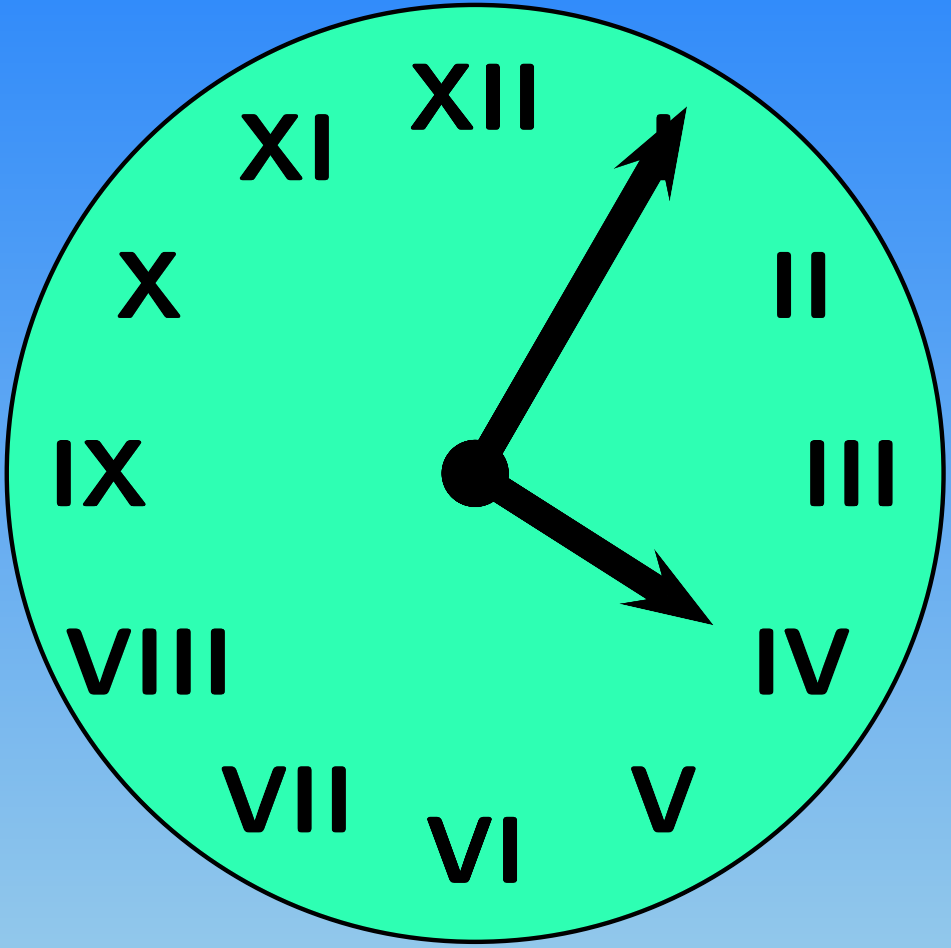


Figure 1: Clock

Animations

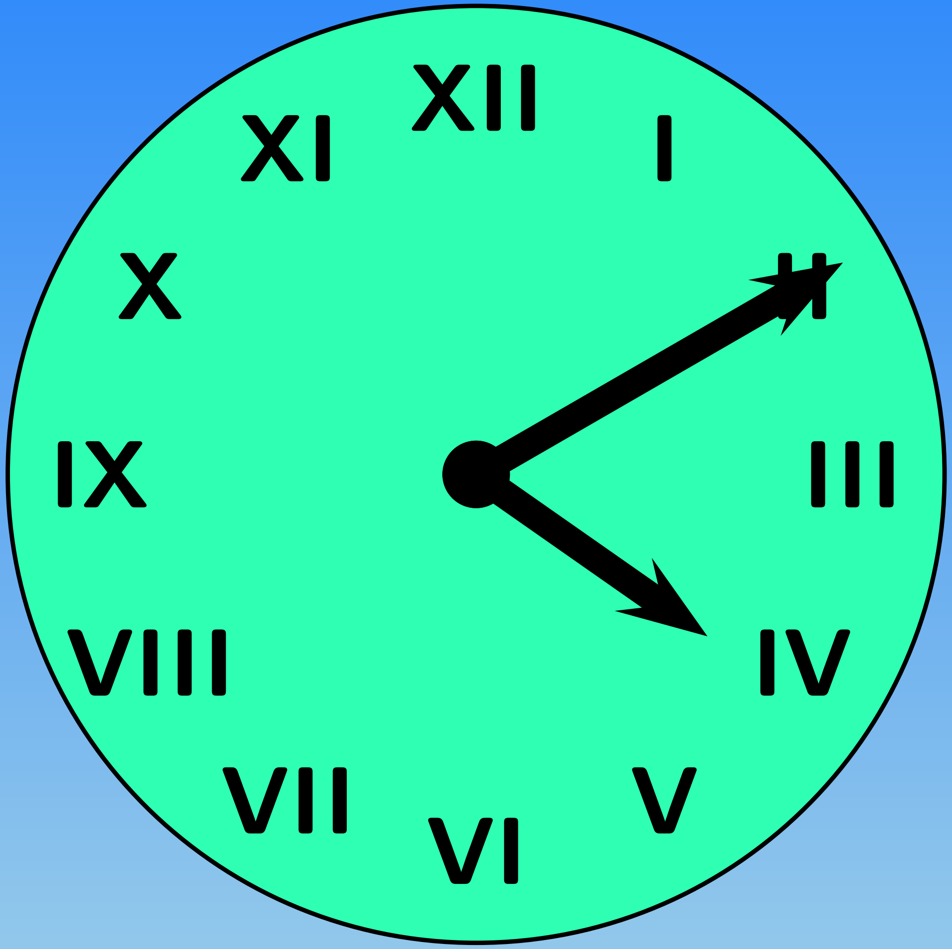


Figure 1: Clock

Animations

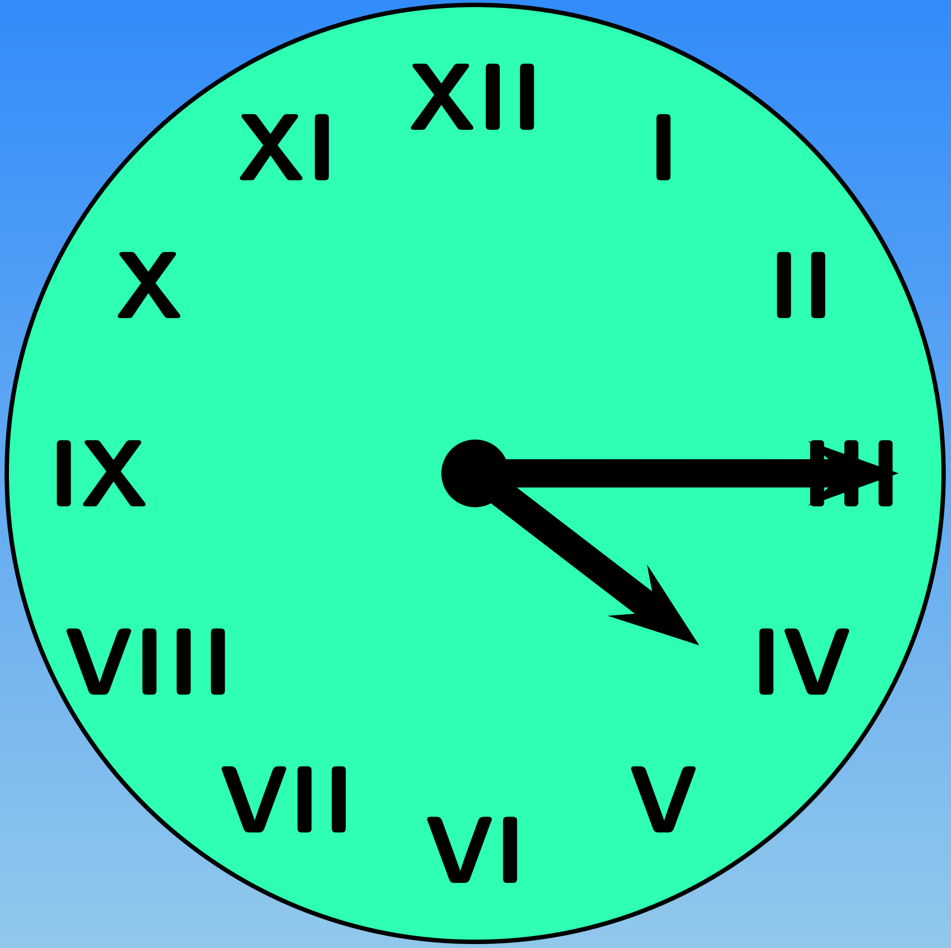


Figure 1: Clock

Animations

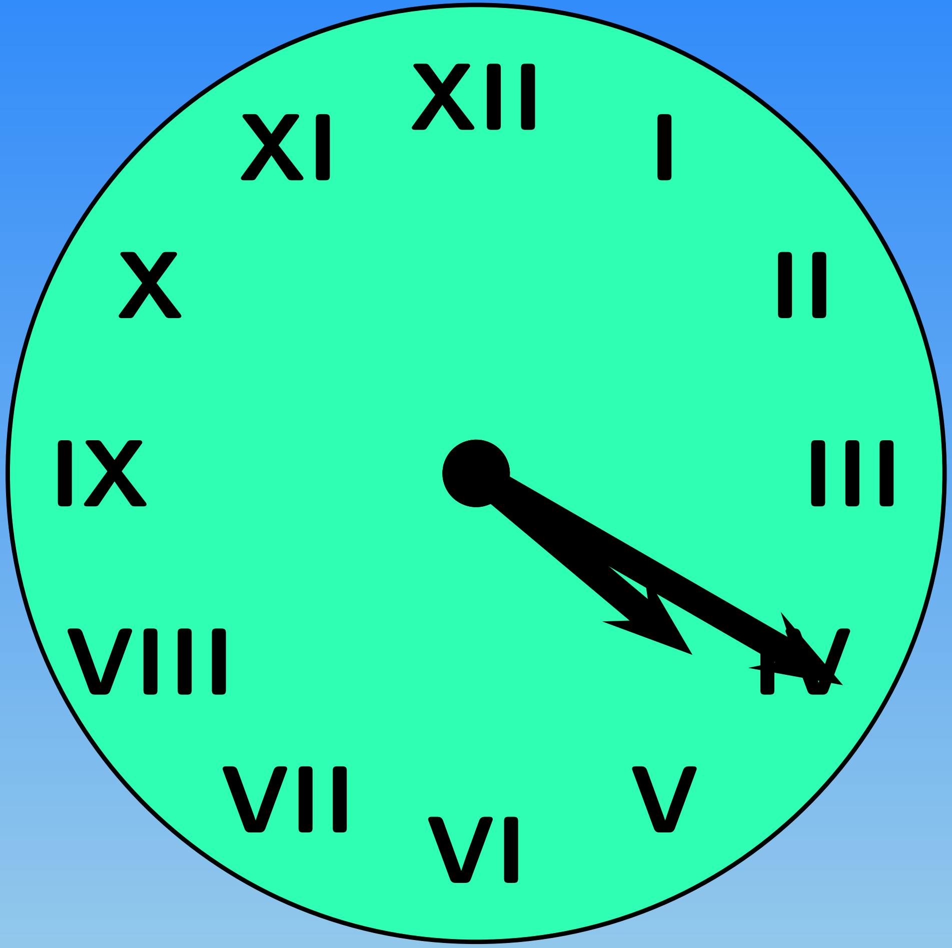


Figure 1: Clock

Animations

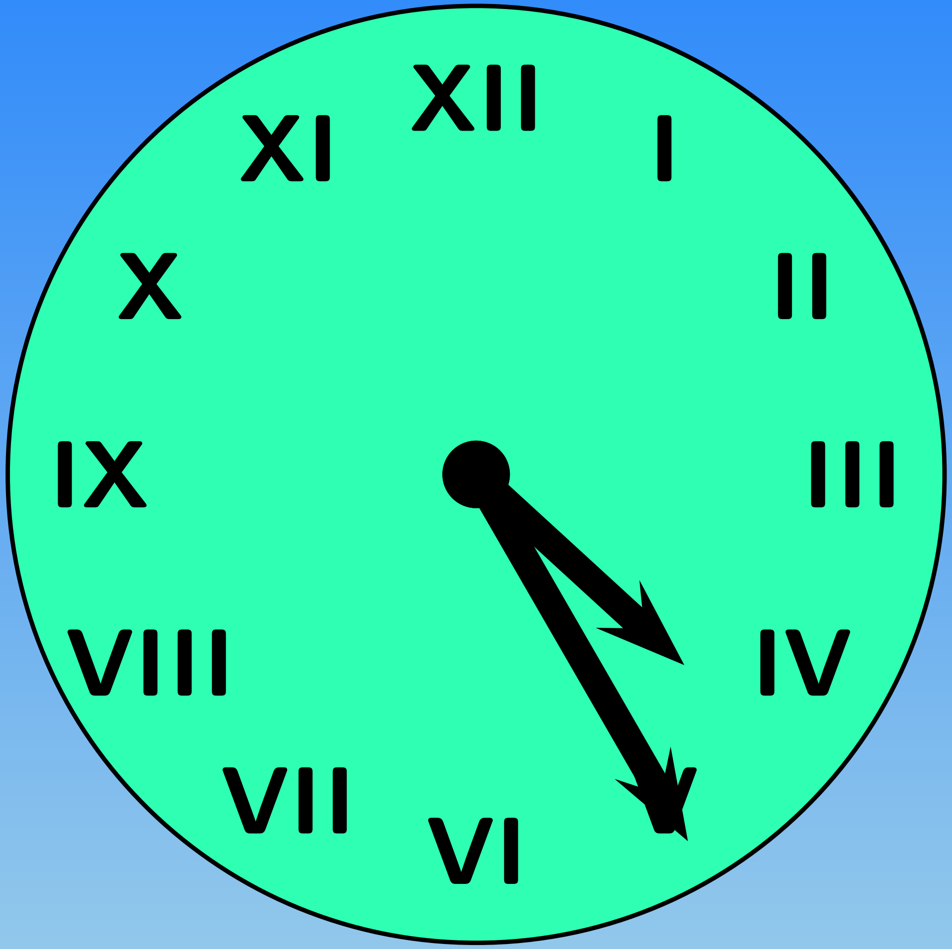


Figure 1: Clock

Animations

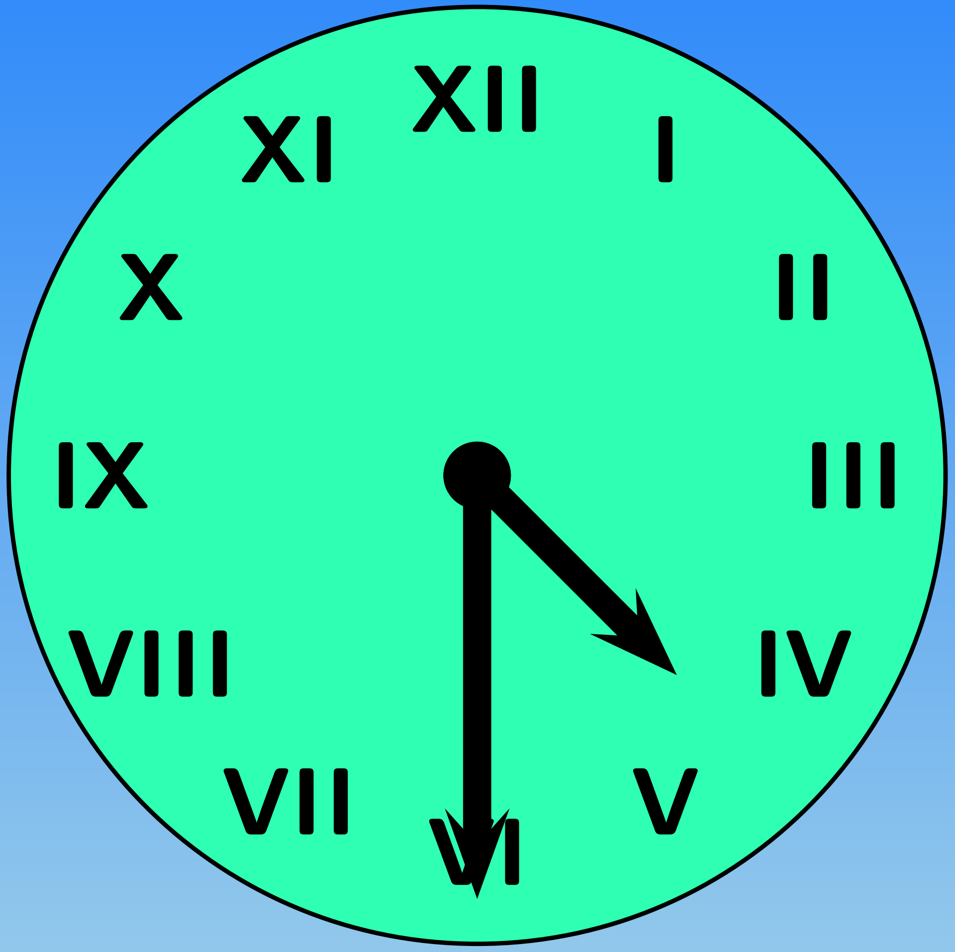


Figure 1: Clock

Animations

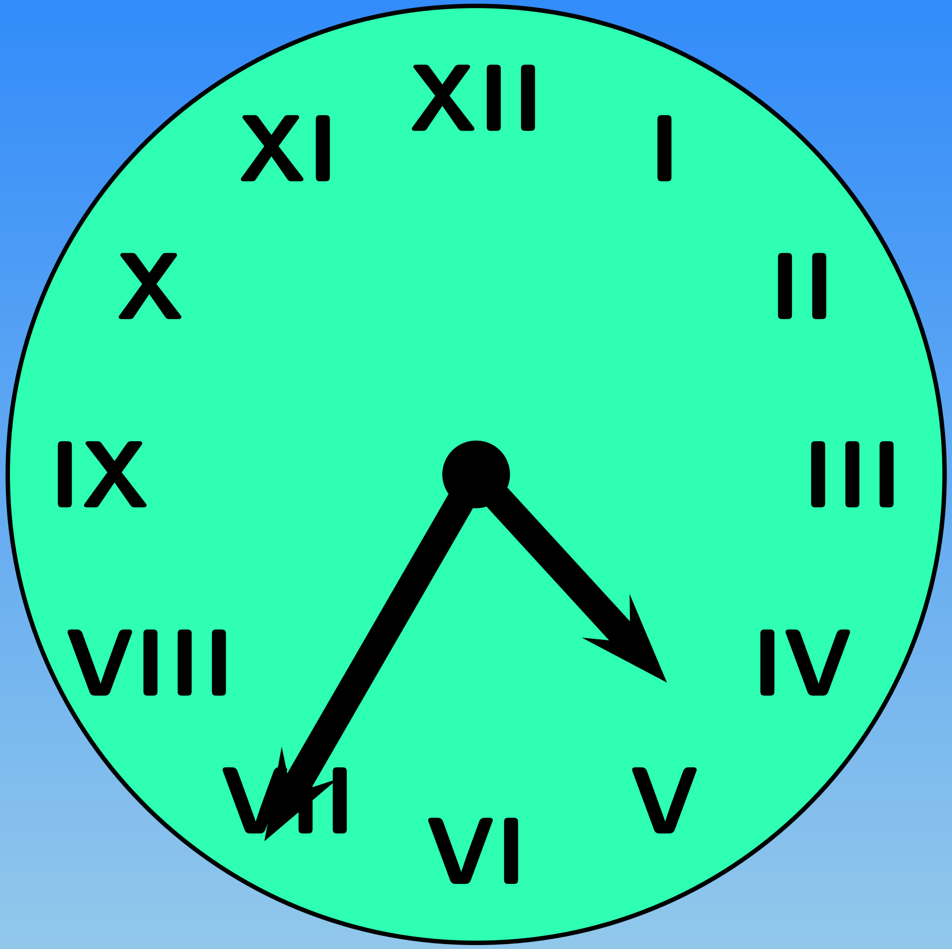


Figure 1: Clock

Animations

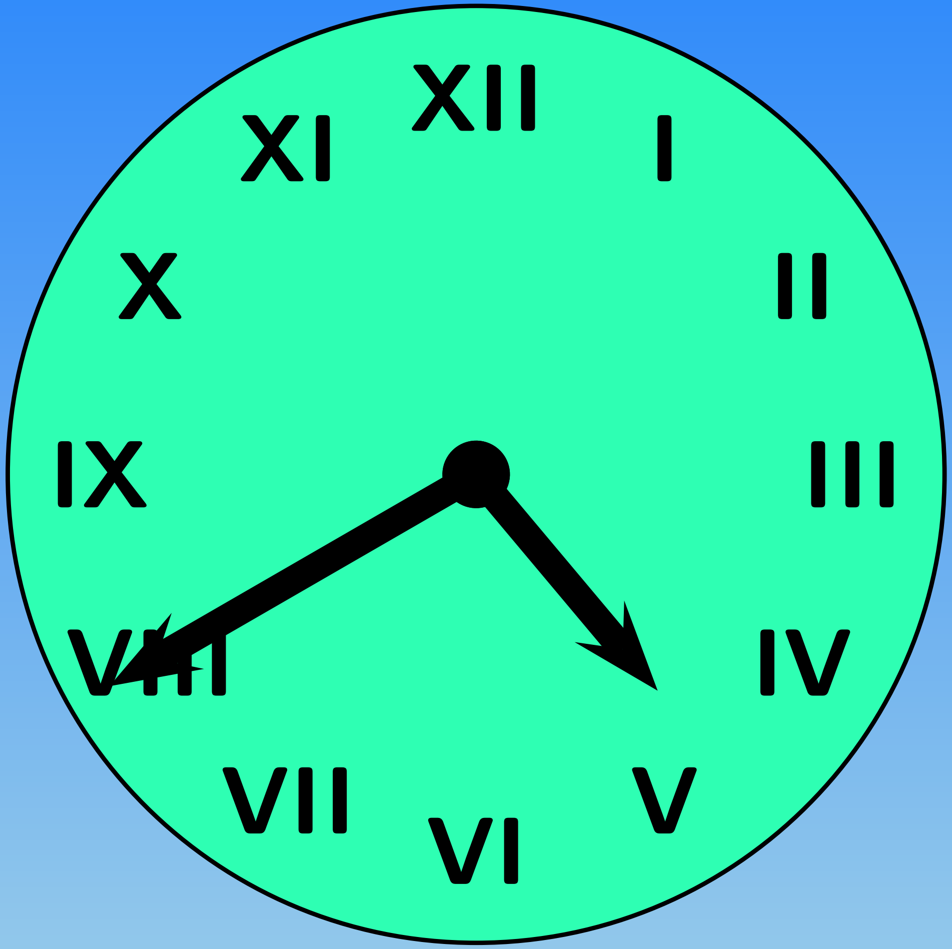


Figure 1: Clock

Animations

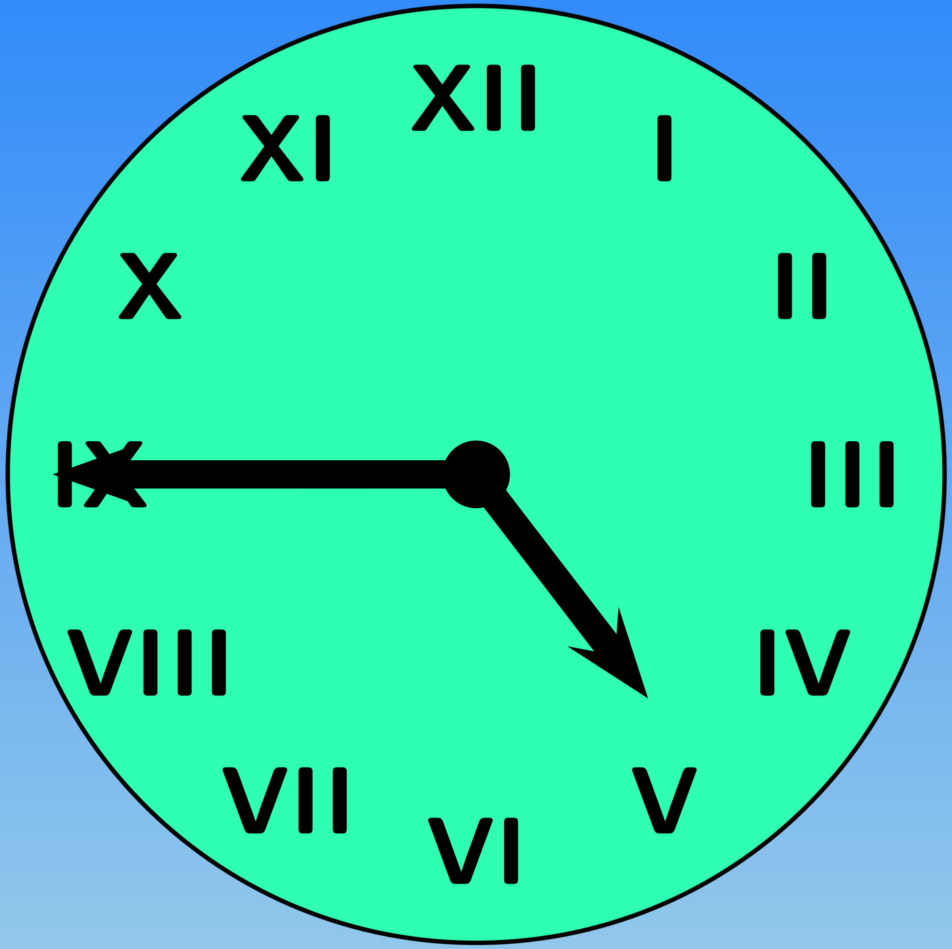


Figure 1: Clock

Animations

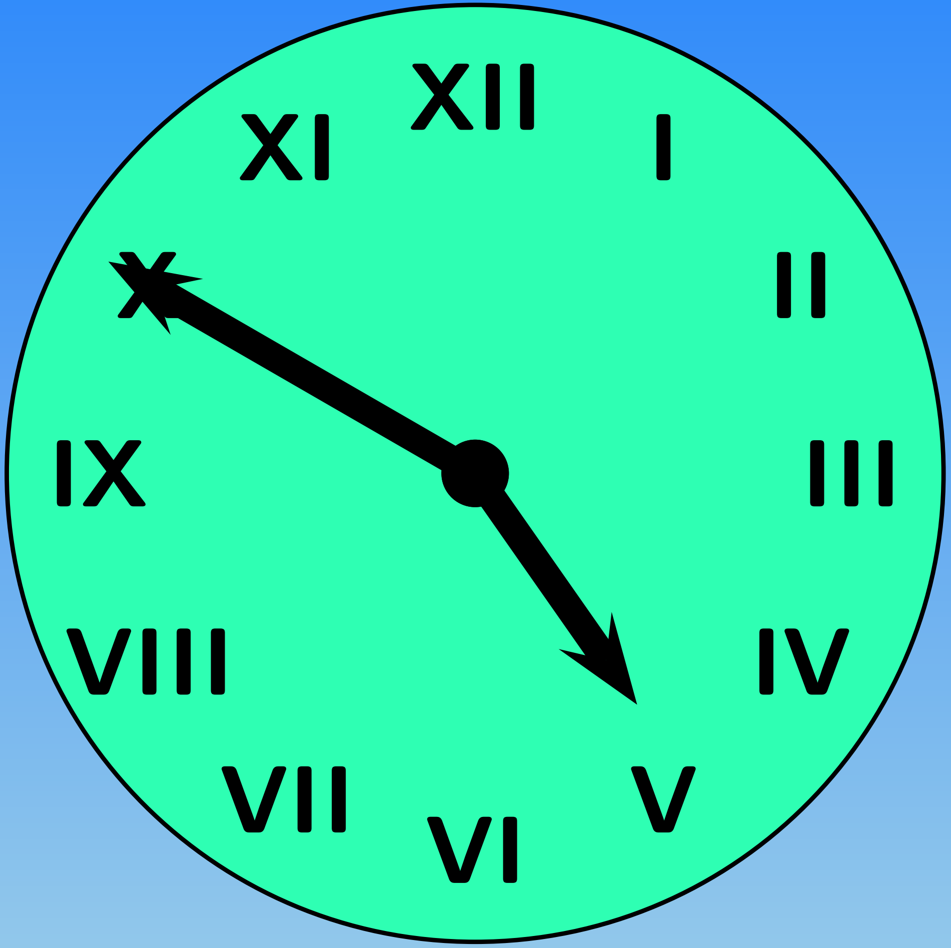


Figure 1: Clock

Animations

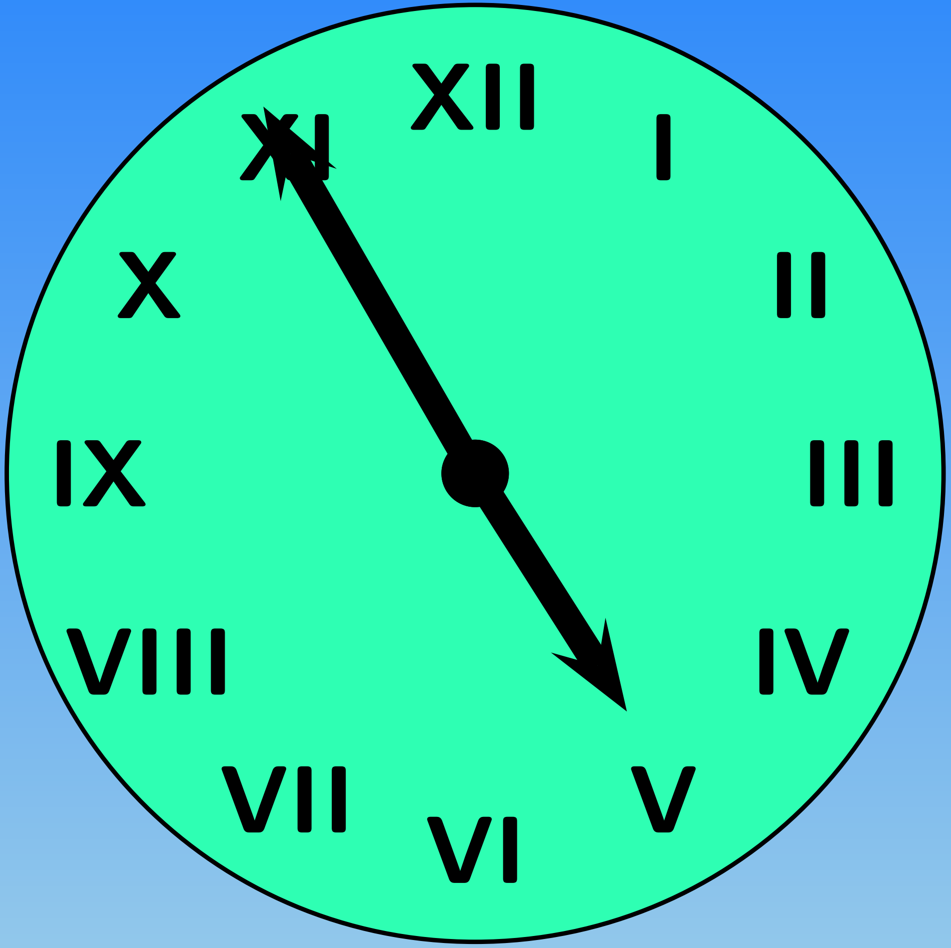


Figure 1: Clock

Animations

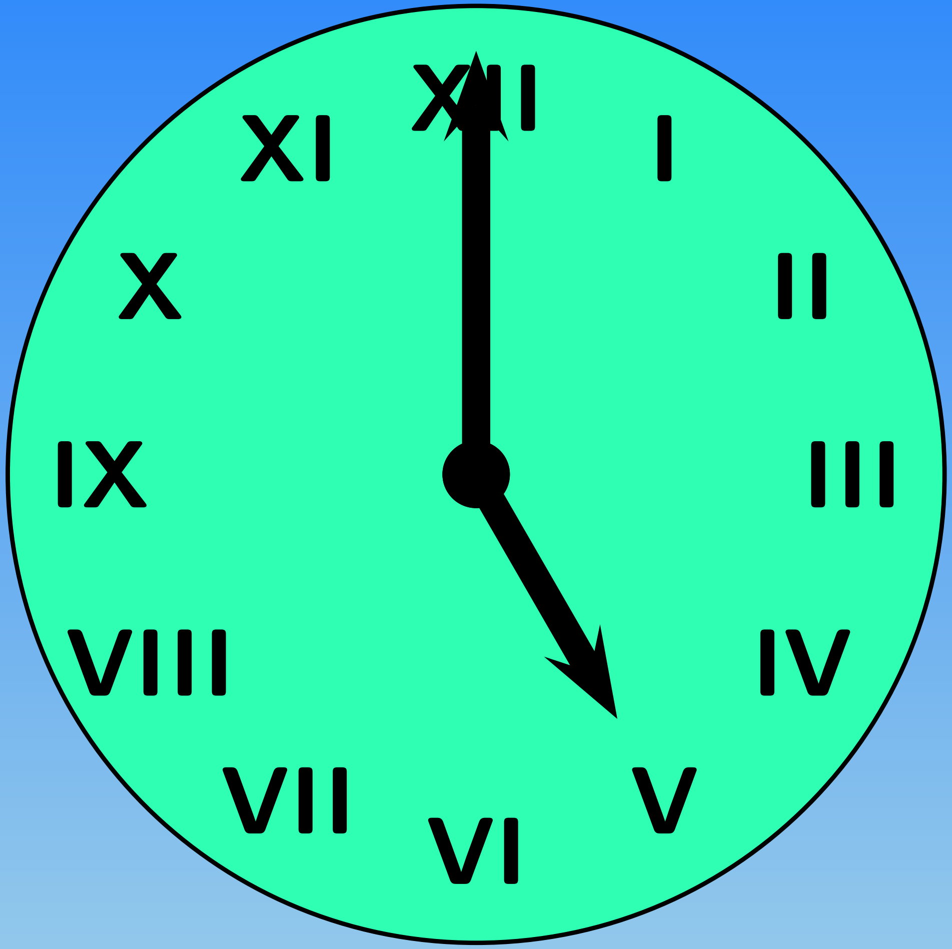


Figure 1: Clock

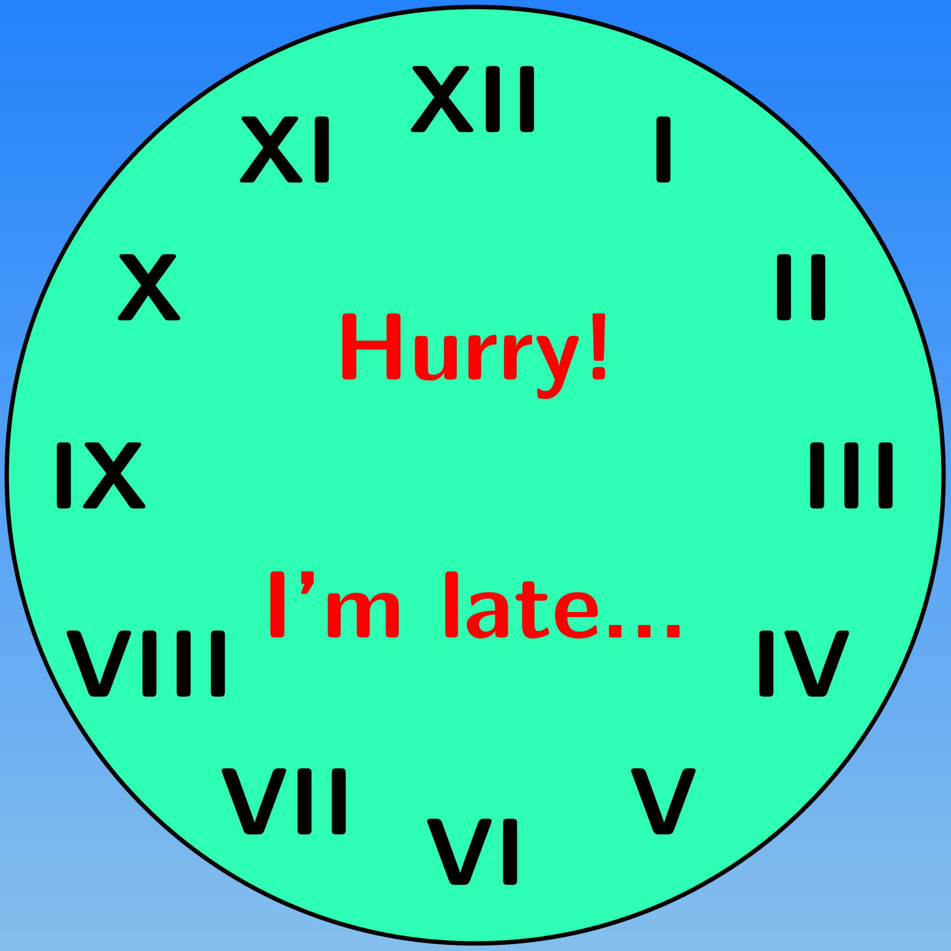


Figure 2: Clock

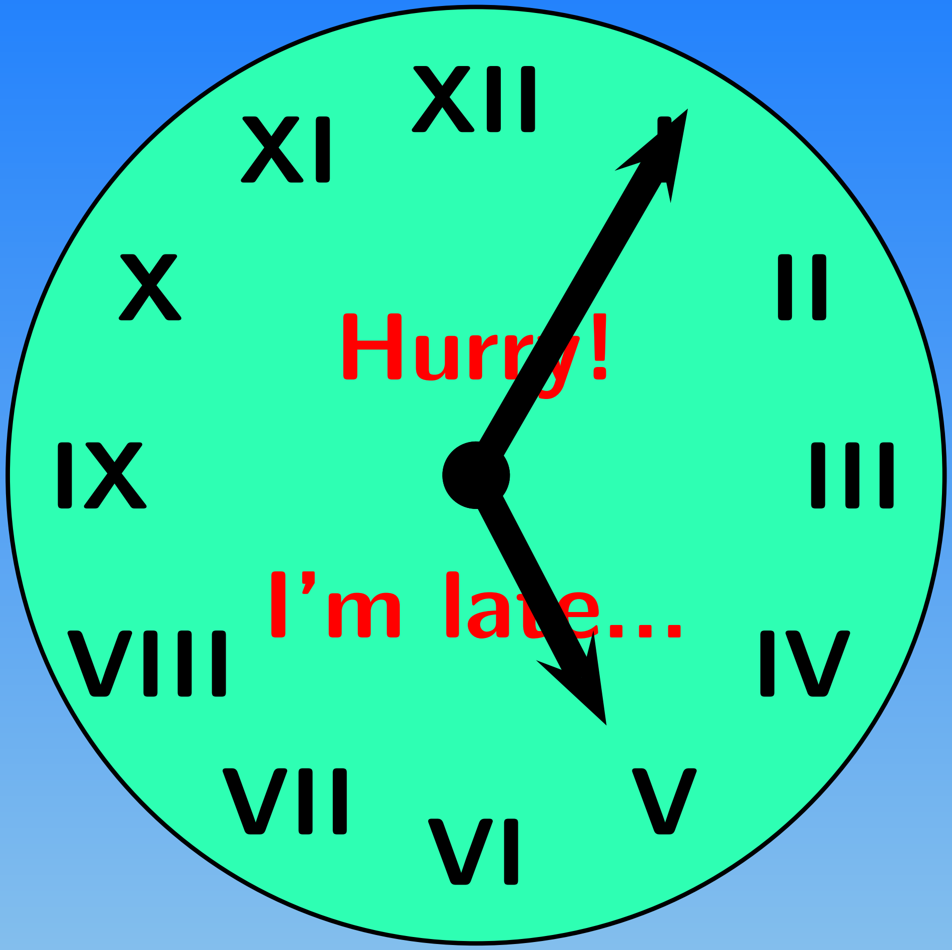


Figure 2: Clock

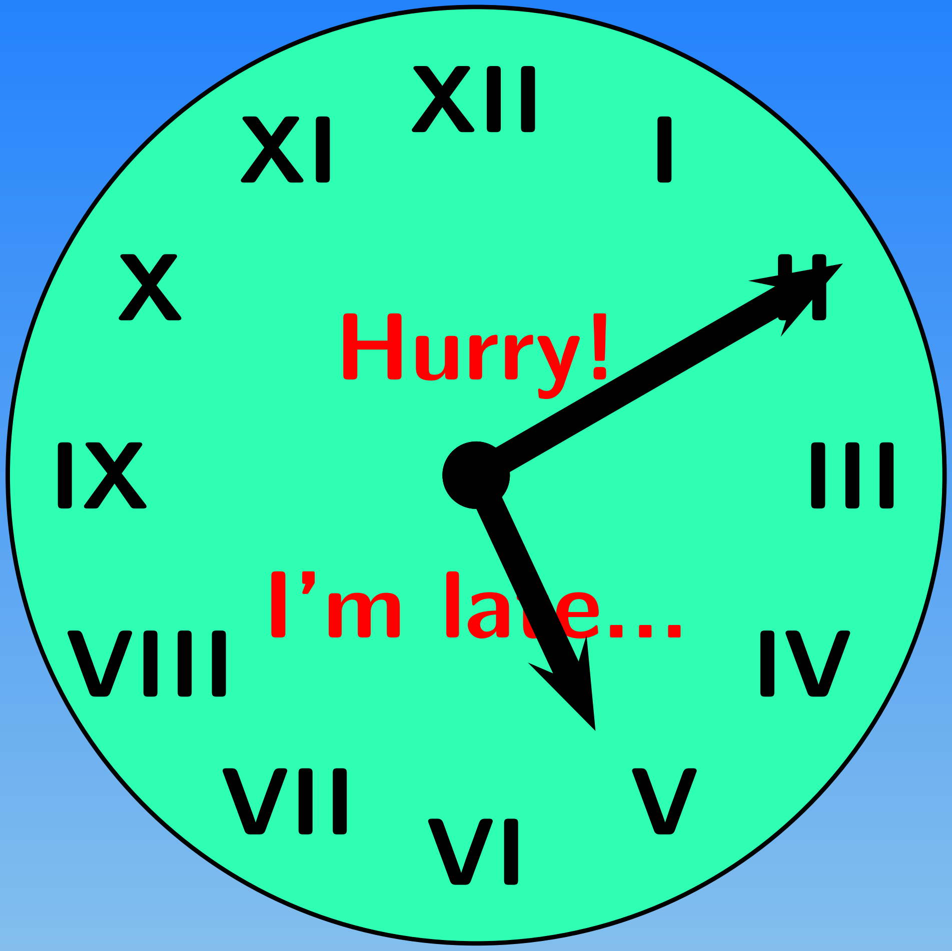


Figure 2: Clock

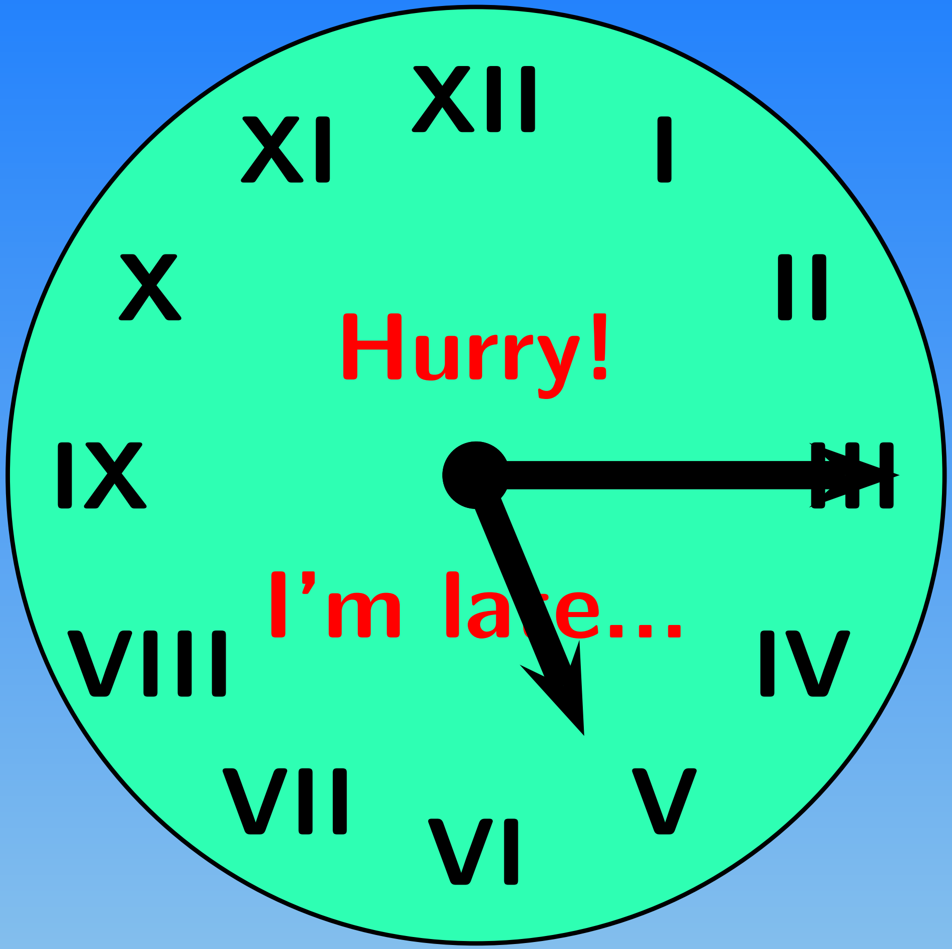


Figure 2: Clock

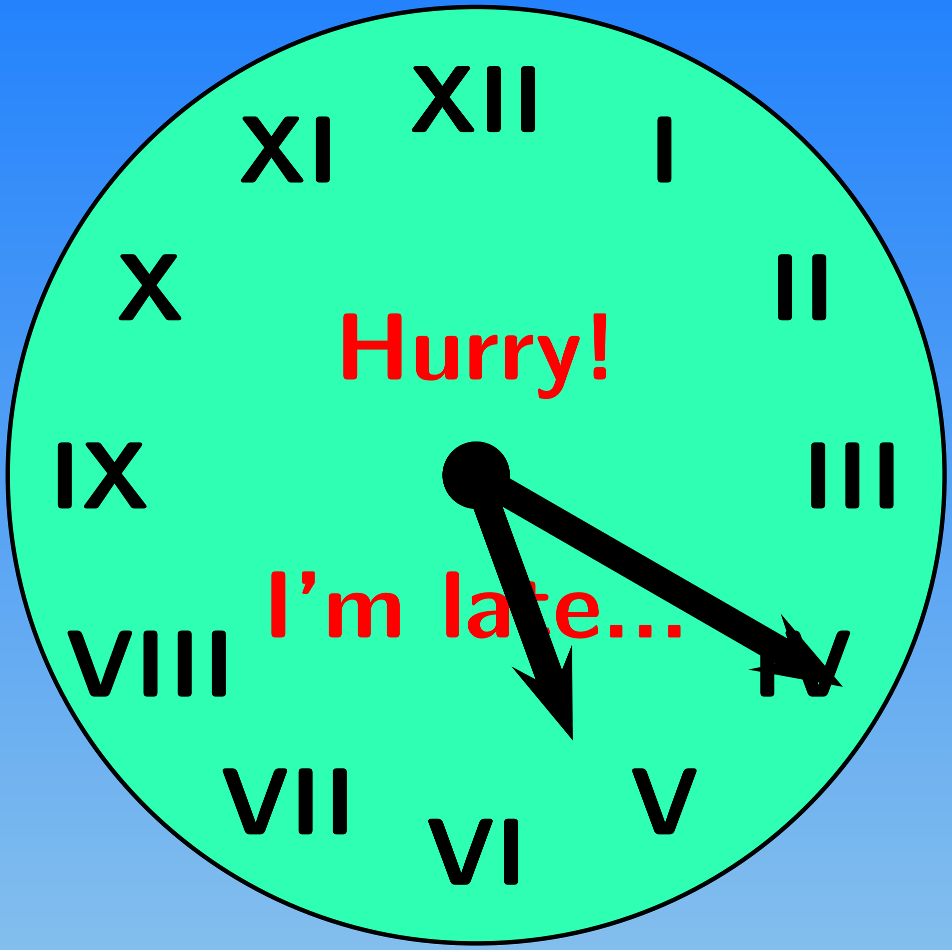


Figure 2: Clock

Figure 2: Clock

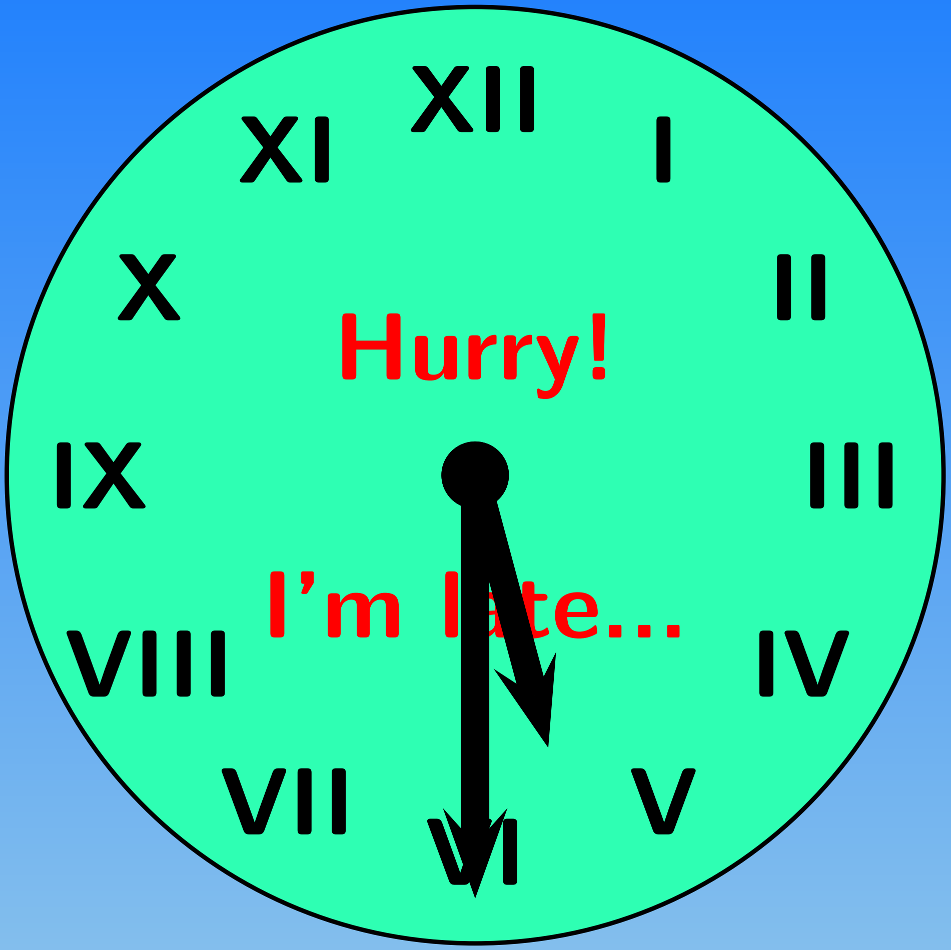


Figure 2: Clock

Figure 2: Clock

Figure 2: Clock

Figure 2: Clock

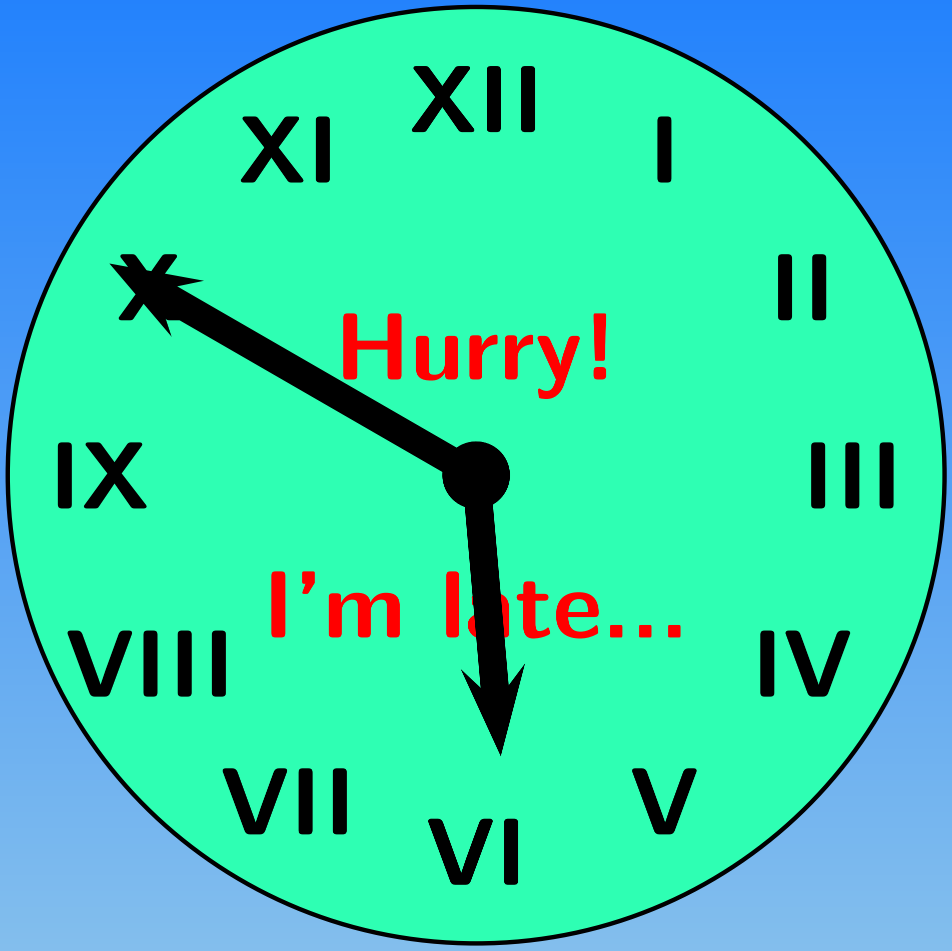


Figure 2: Clock

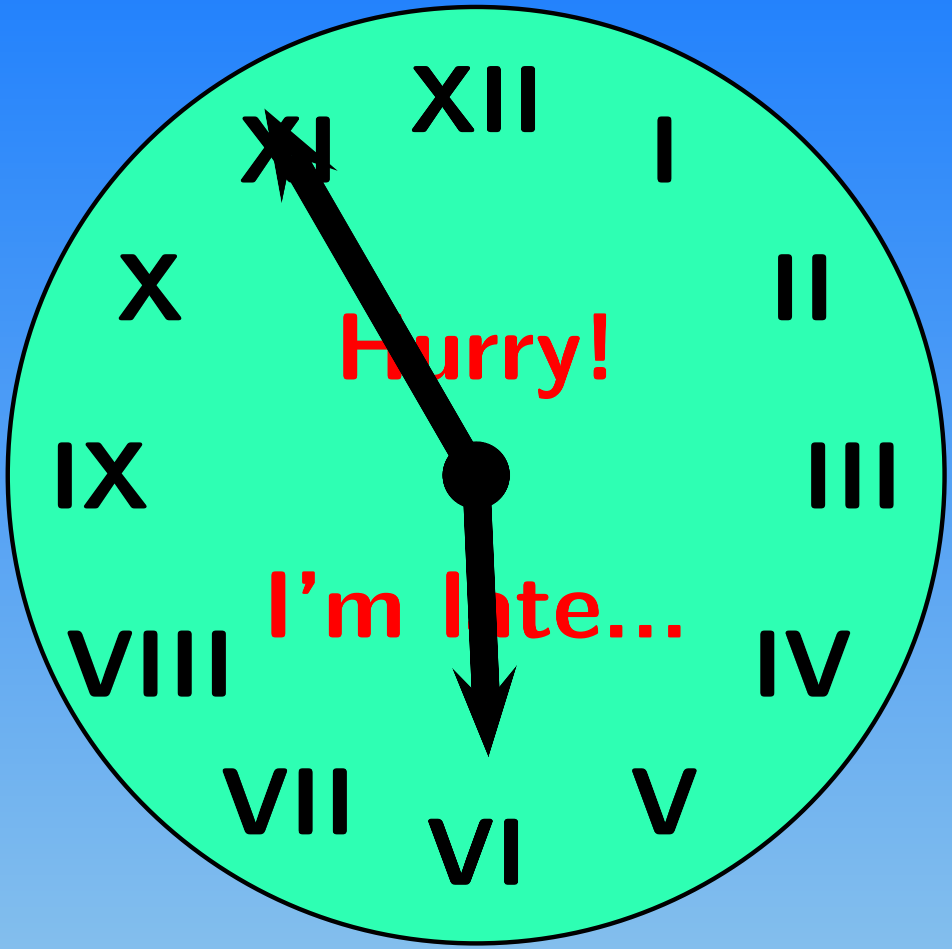


Figure 2: Clock

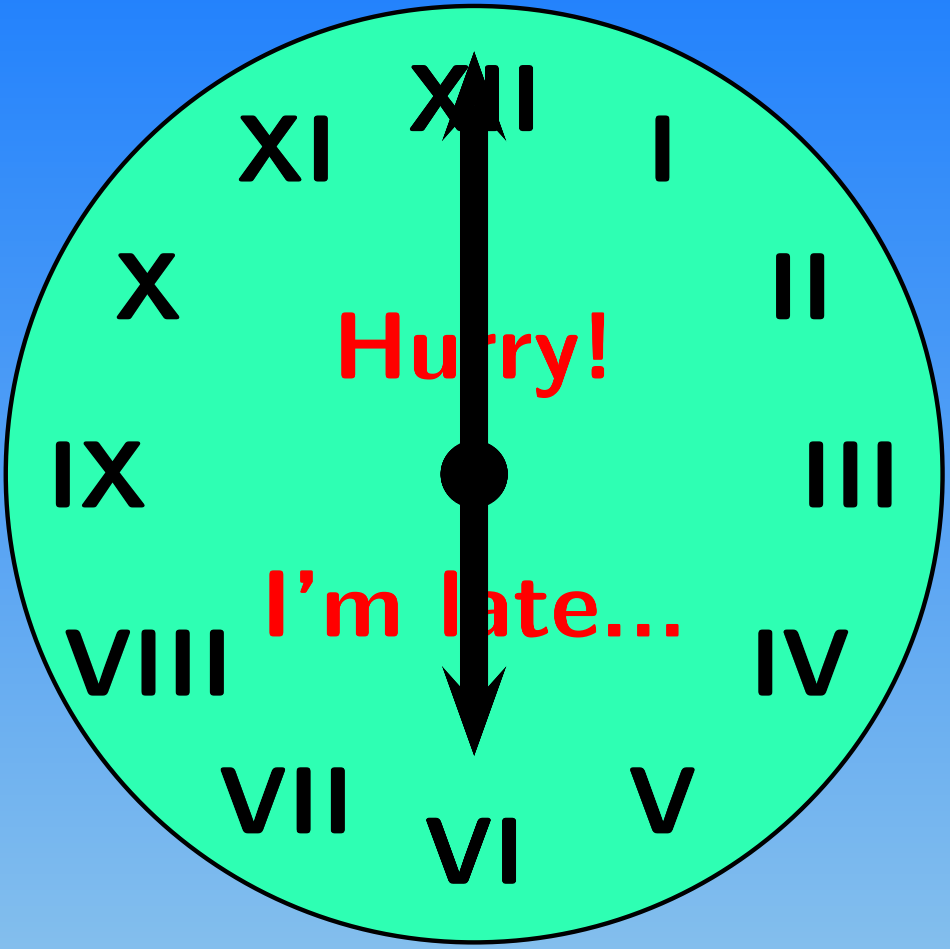


Figure 2: Clock

Document compiled at: 0h 55m

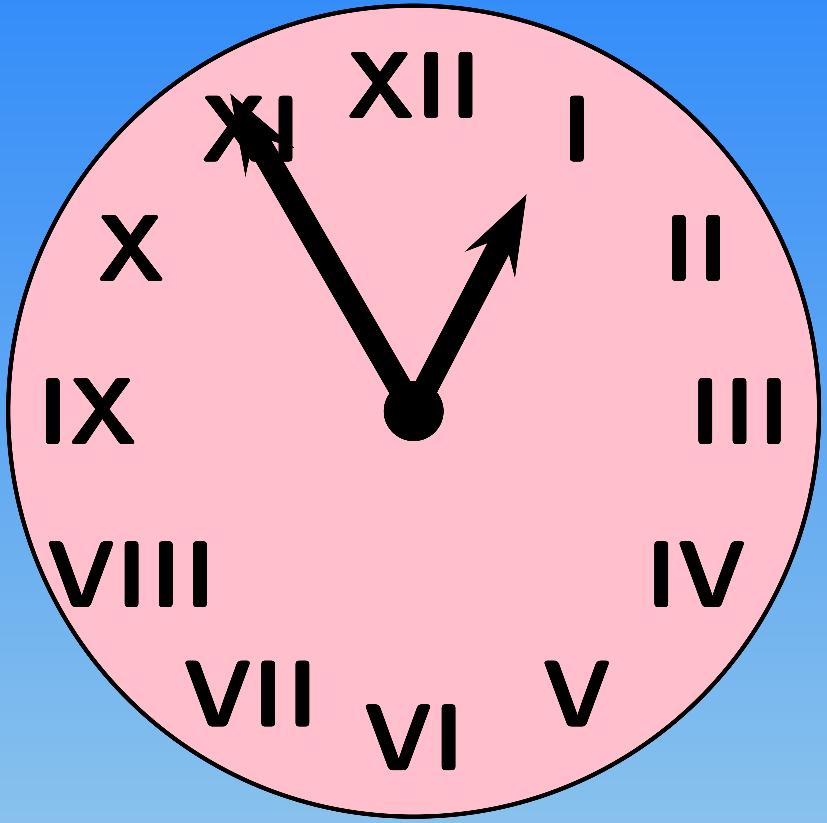


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 01s

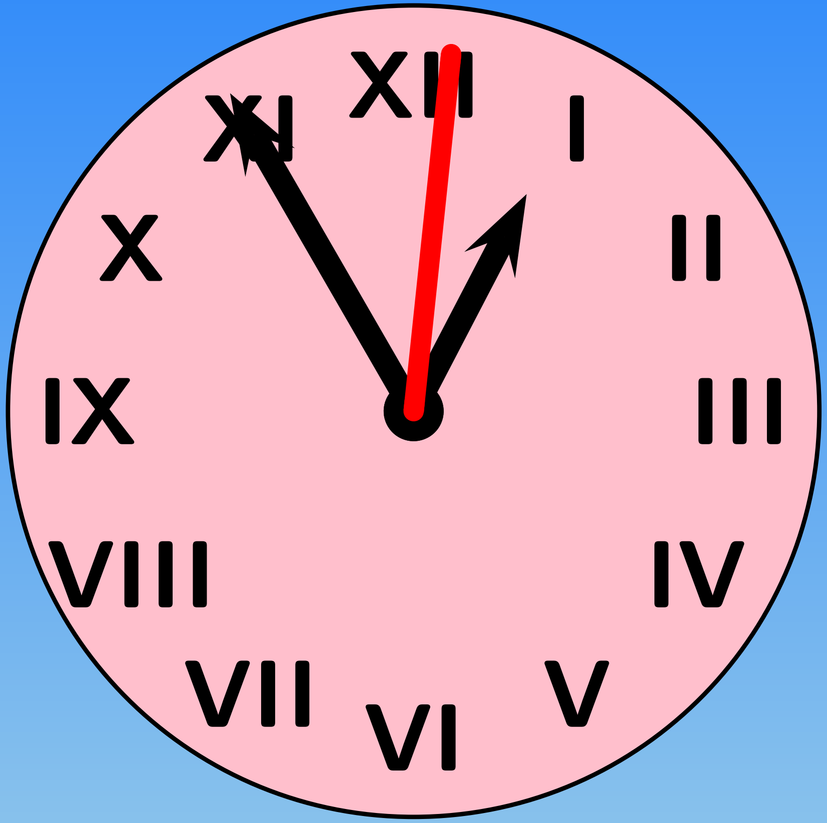


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 02s

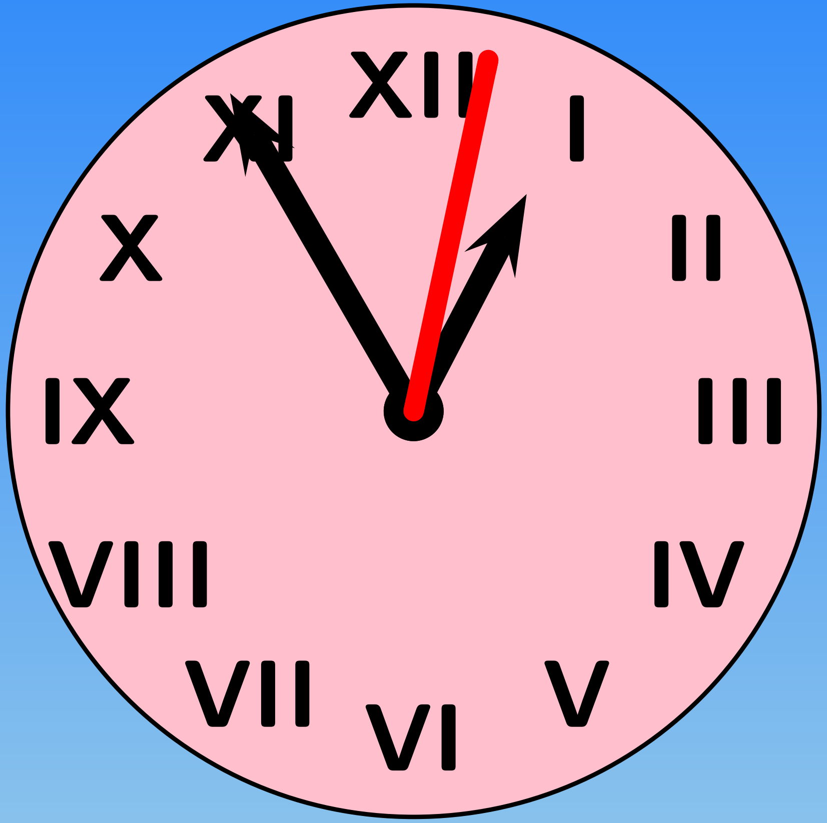


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 03s

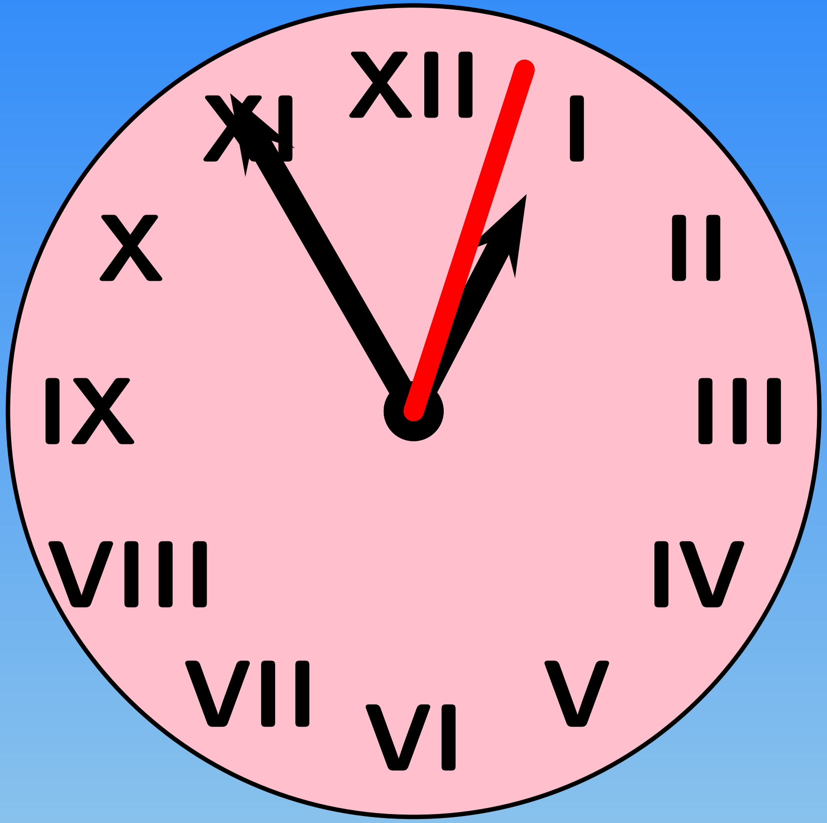


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 04s

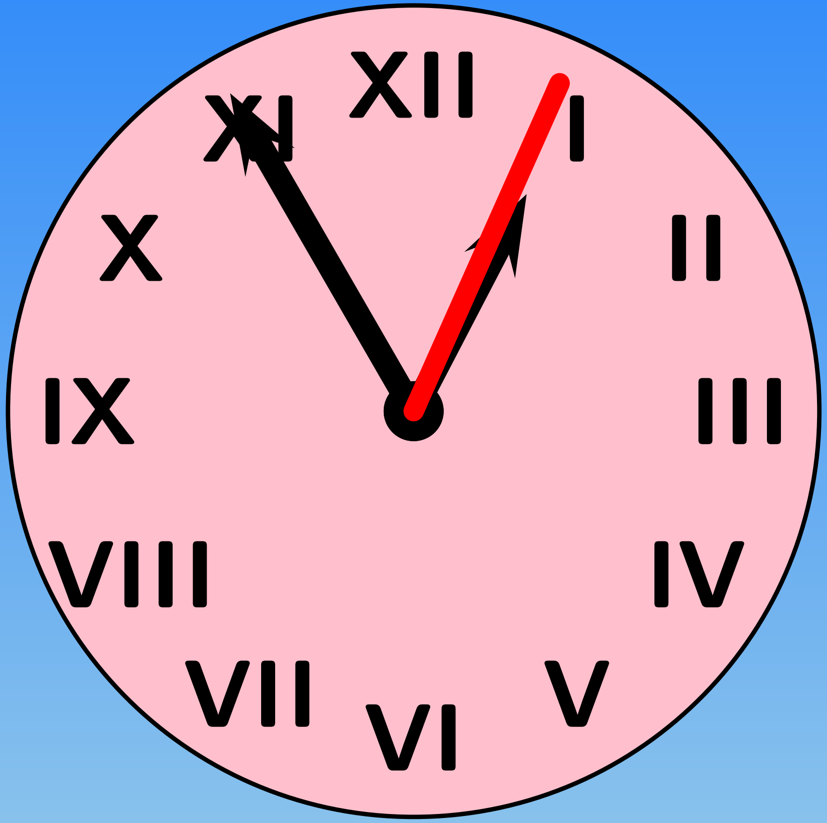


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 05s

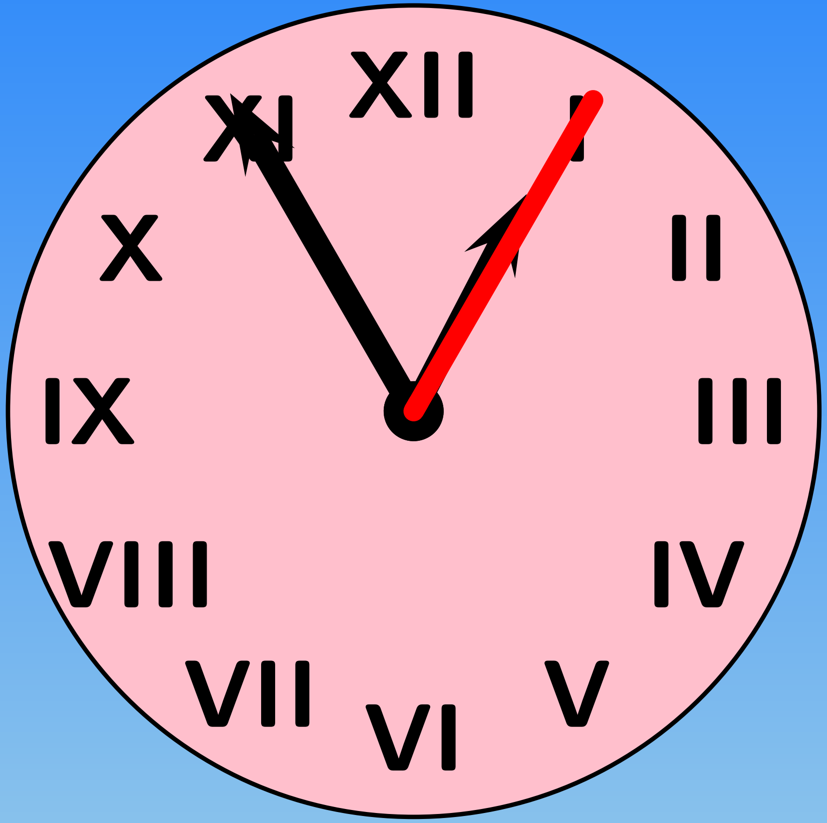


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 06s

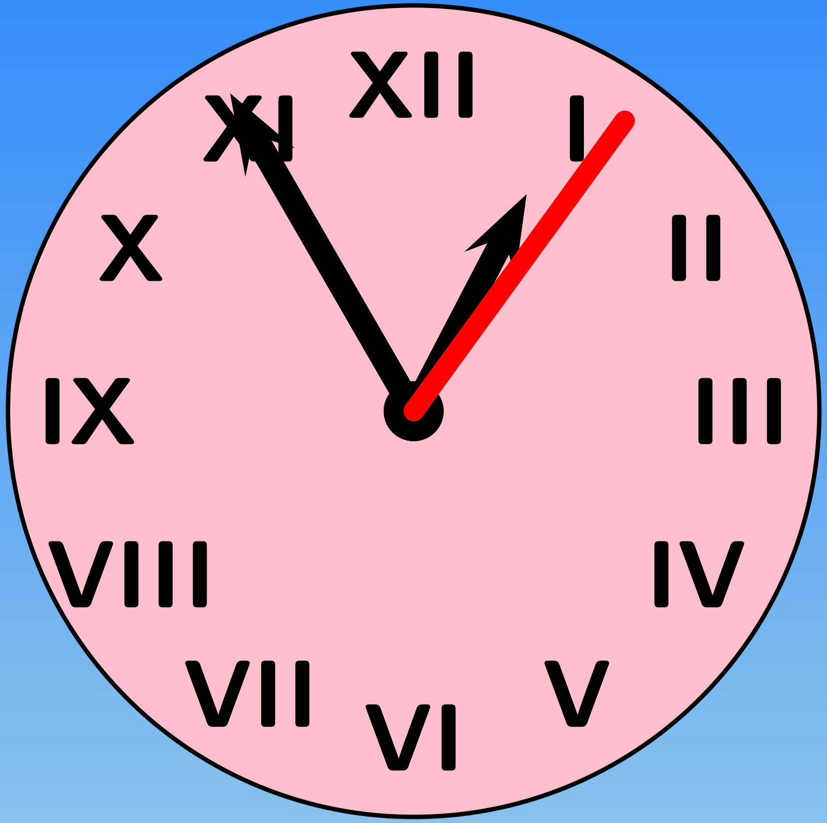


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 07s

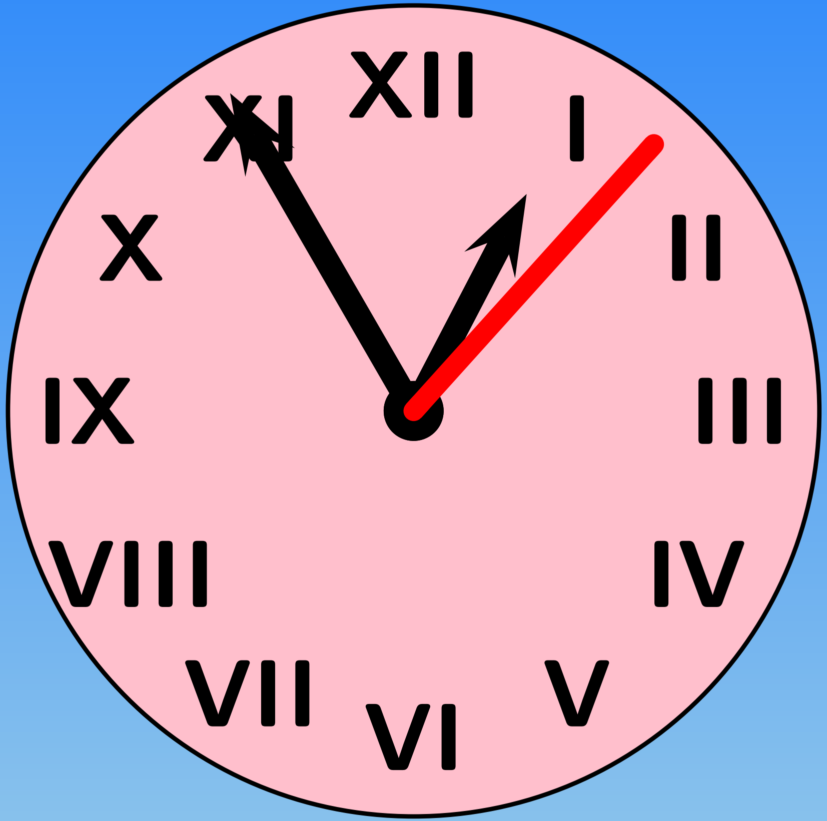


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 08s

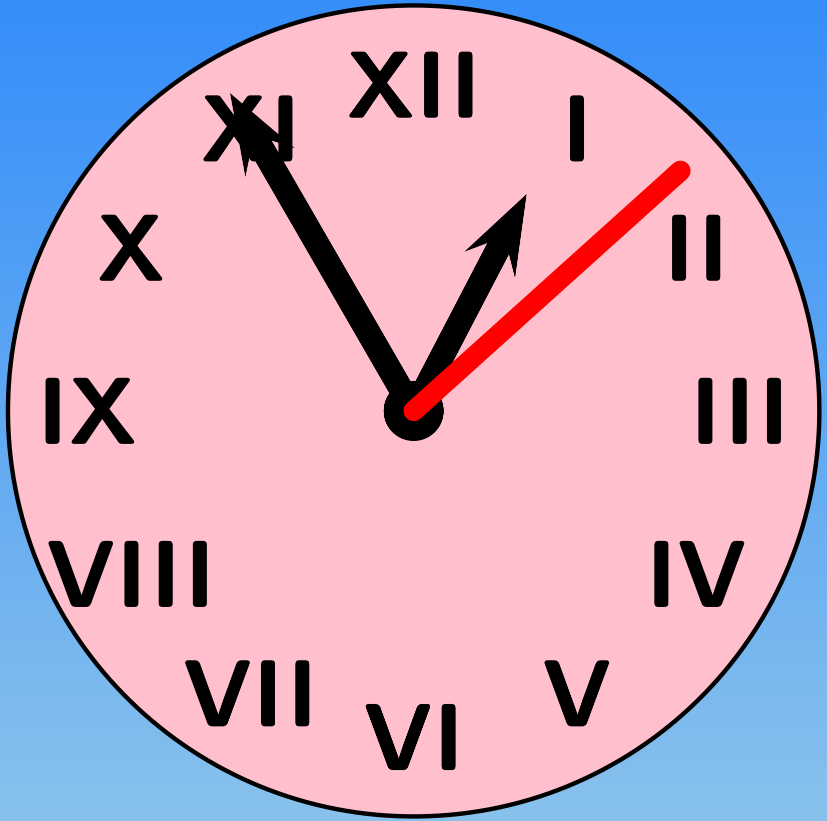


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 09s

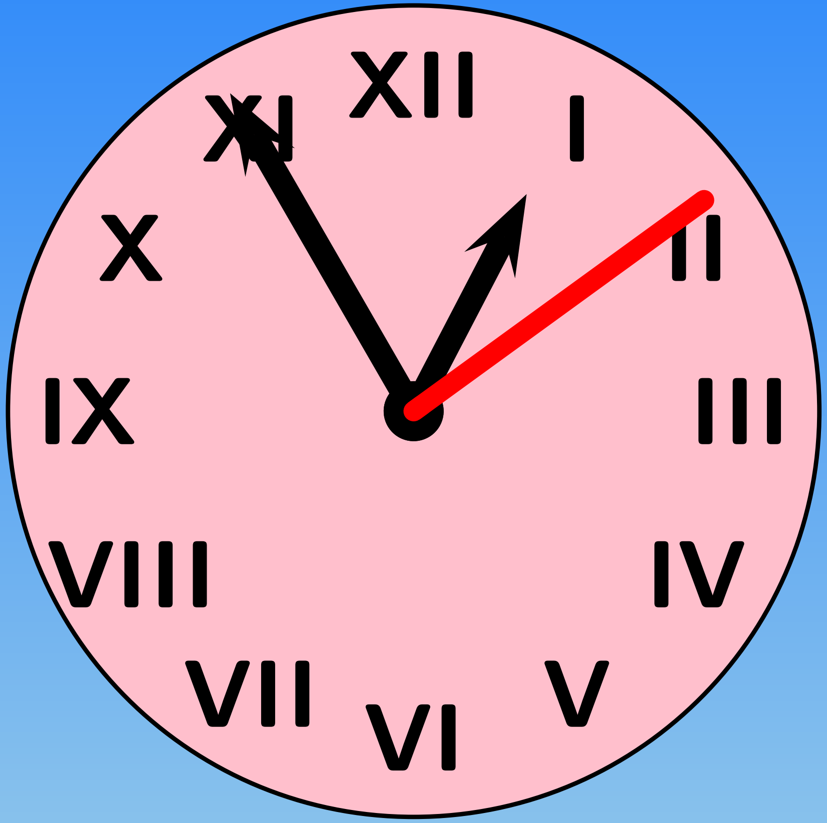


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 10s

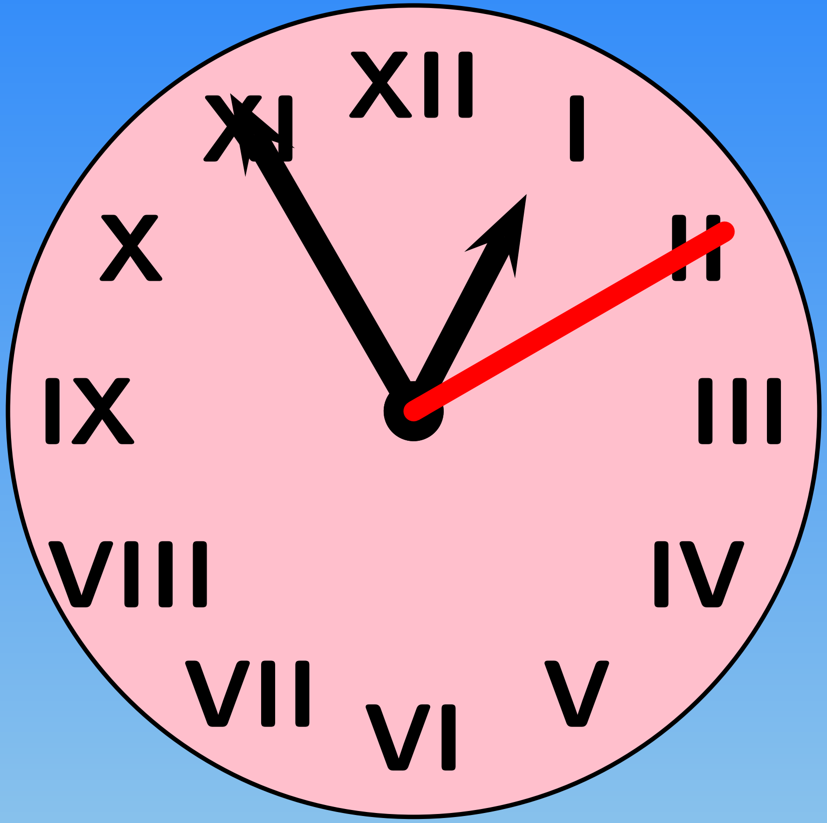


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 11s

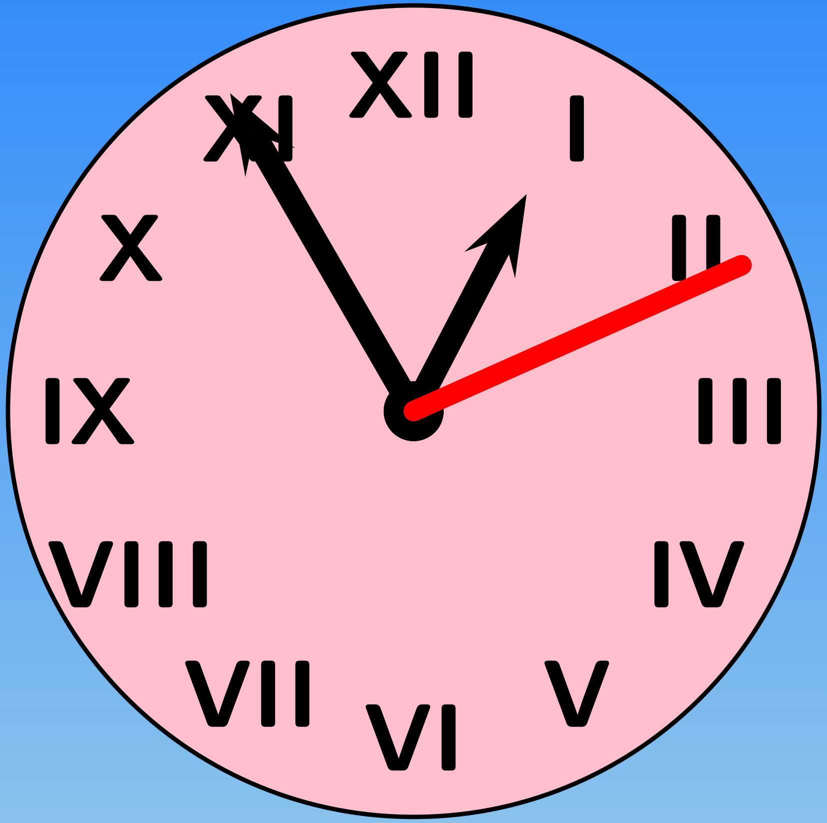


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 12s

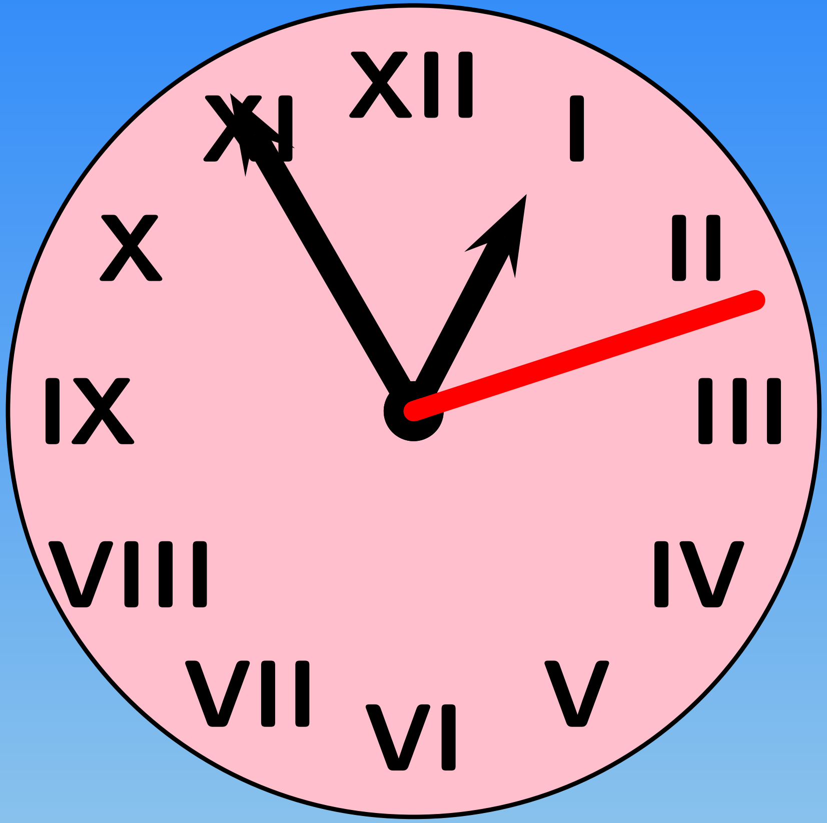


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 13s

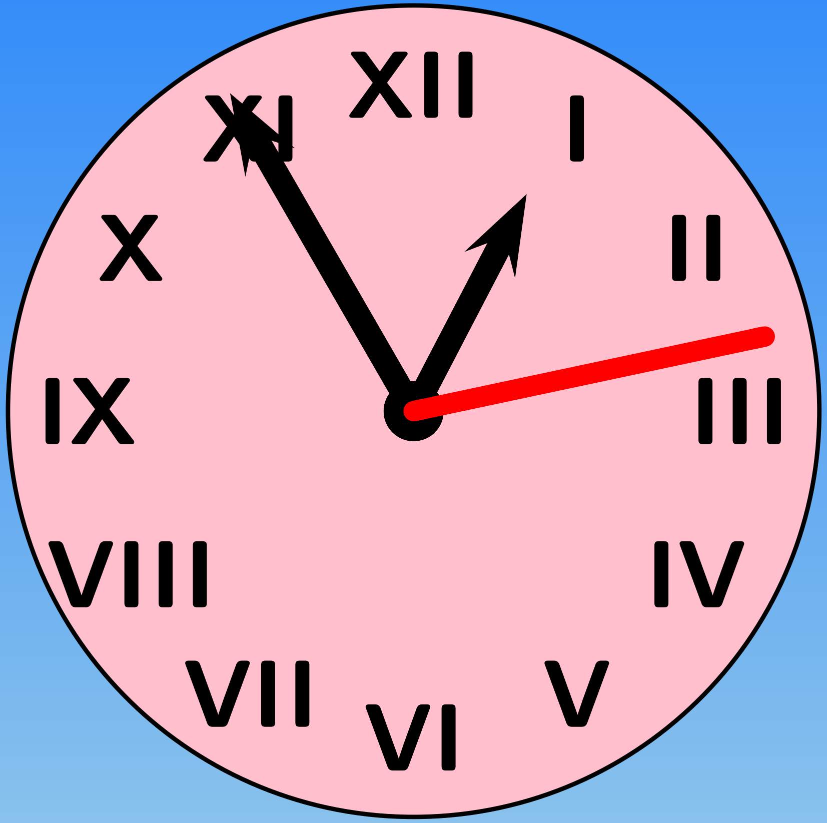


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 14s

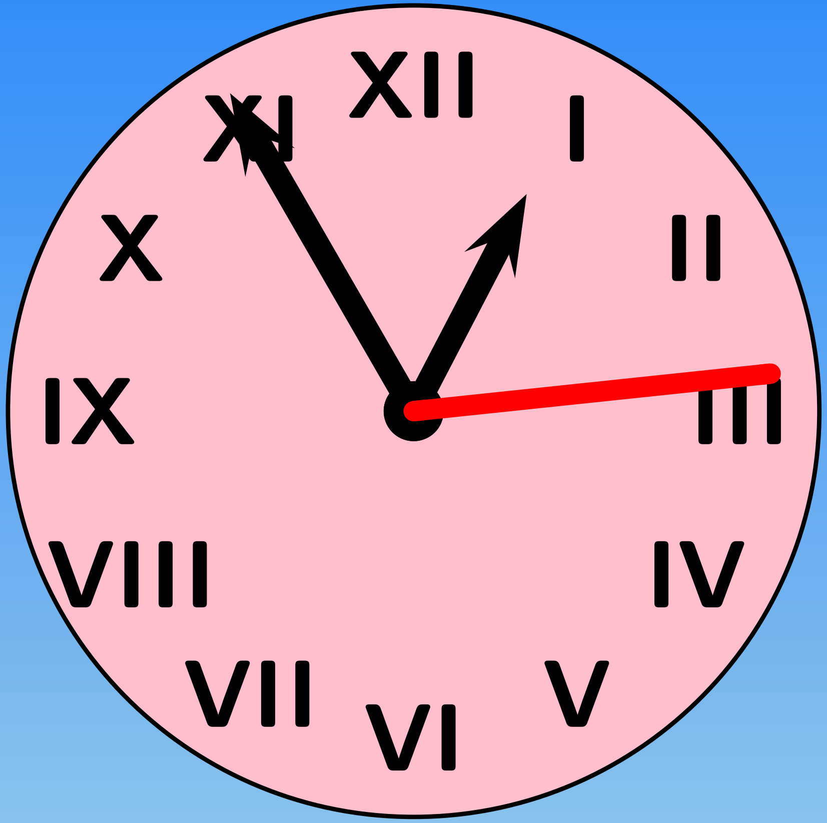


Figure 3: Clock with split-second hand

Document compiled at: 0h 55m 15s

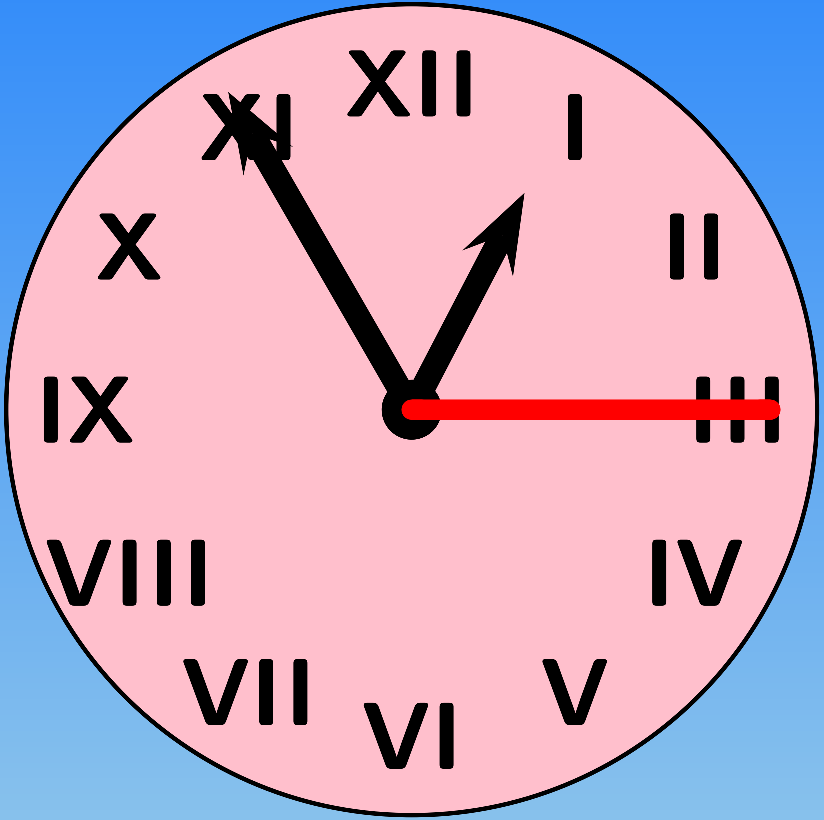


Figure 3: Clock with split-second hand

Example: Building of a regular polygon of seventeen sides

Movies, Sound, External Application

PDF supports the following sound and movie formats:

		Format	Underlying Application
movie	–	`avi` **format**	**Windows Media Player**
		`mov` **format**	**Quick Time Player**
sound	–	`wav` **format**	**Windows Media Player**
		`aif` **format**	**Windows Media Player**

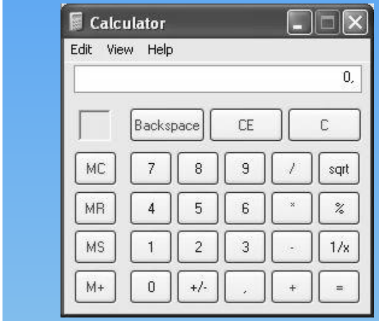
Movie in the `mov` format

Movie in the `avi` **format**

Sound in the `wav` format:

command

Calling an external application:



Electronic Testing – Examples

AcroTEX	–	Donald Story	example1
		University of Akron, USA	example2
MacQTEX	–	Ross Moore	example1
		Macquarie University, Australia	example2
			example3
			example4
			example5