

The Choose-Your-Own-Adventure Calculus

Tomas Petricek ✉ 

Charles University, Prague, Czechia

Jan Liam Verter ✉

Charles University, Prague, Czechia

Mikoláš Fromm ✉

Charles University, Prague, Czechia

Abstract

Some of the most remarkable results in mathematics reveal surprising connections between different branches of the discipline. The aim of this paper is to point out a modest, but still remarkable, similarity between a range of different interactive programming systems.

use a simple formal mathematical model that we call *the choose-your-own-adventure calculus* to
todo

2012 ACM Subject Classification Software and its engineering → General programming languages

Keywords and phrases Interactive programming systems, Type providers, Proof assistants

1 Introduction

Multiple interactive programming systems, ranging from code editors for object-oriented programming languages to data exploration systems and interactive proof assistants, exhibit a remarkably similar pattern of interaction. They offer the user, who can be a programmer, a data scientist or a proof writer, a range of choices that the user can select from in order to complete their program, script or proof. The user can initiate the interaction iteratively, using it to create and refine a larger part of their program.

There are subtle differences between different implementations of the general pattern. In some systems, the resulting source code will contain a trace of the choices made by the user. For example, when choosing an item from a list of class members, the code will contain the member name. In some systems, the interaction results in a block of code that can be included in the source file, but does not include a trace of the interaction. For example, invoking a proof search or case split in Idris [3] constructs a well-typed program, but leaves no trace of the command used to construct it. The nature of the generated options also varies. The list of choices may include all possible options that are valid at a given location, or it may list only a subset of the valid options. In some cases, it may also include incorrect options as, for example, in auto-completion for dynamic languages [4].

The aim of this paper is to formally capture the recurring interaction pattern:

1. We motivate the formalism by reviewing four different systems that implement a variation on the interaction pattern. These include type providers for data access in F# [11], type providers for data exploration in The Gamma [9, 7], AI assistants for semi-automated data wrangling [10] and tooling for interactive proof assistants [2, 3, 12] (Section 2).
2. We introduce the *choose-your-own-adventure calculus*, which is a small formal structure that models an interactive system where a user constructs a program by repeatedly choosing from a list of options offered by the system (Section 3).
3. The calculus allows us to make the aforementioned subtle differences precise. We define the notions of *correctness* and *completeness* for the choose-your-own-adventure calculus. To distinguish the different ways of embedding the interactions in the edited programs, we also formally define *base-level* and *meta-level* integration of the system.
4. We show that various programmer assistance tools, such as search and AI-based recommendations can be built on top of the primitives offered by the calculus, showing how the choose-your-own-adventure calculus supports of transfer of ideas across different kinds of interactive programming systems.

The main contribution of this paper is conceptual rather than technical. Programming language researchers have long focused on *programming languages* rather than on a more general notion of interactive *programming systems* [5, 6]. However, programming language theory methods can be equally useful for studying interactive systems [1]. The choose-your-own-adventure-calculus is another step in this direction. It brings to light a pattern that is perhaps not surprising, but is very easy to overlook until it is given a name.

2 Motivation

3 Formal model

4 Examples

5 Properties

6 Applications

[6] [8]



References

- 1 Michael D. Adams, Eric Griffis, Thomas J. Porter, Sundara Vishnu Satish, Eric Zhao, and Cyrus Omar. Grove: A bidirectionally typed collaborative structure editor calculus. *Proc. ACM Program. Lang.*, 9(POPL), January 2025. doi:10.1145/3704909.
- 2 Thorsten Altenkirch, Veronica Gaspes, Bengt Nordström, and Björn von Sydow. A user's guide to alf. Technical report, Chalmers University of Technology, Sweden, 1994. Unpublished Draft. URL: <https://people.cs.nott.ac.uk/psztxa/publ/alf94.pdf>.
- 3 Edwin Brady. *The Idris Programming Language*, pages 115–186. Springer International Publishing, Cham, 2015. doi:10.1007/978-3-319-15940-9_4.
- 4 Damian Frölich and L. Thomas van Binsbergen. On the soundness of auto-completion services for dynamically typed languages. In *Proceedings of the 23rd ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences*, GPCE '24, page 107–120, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3689484.3690734.
- 5 Richard P. Gabriel. The structure of a programming language revolution. In *Proceedings of the ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, Onward! 2012, page 195–214, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2384592.2384611.
- 6 Joel Jakubovic, J. Edwards, and T. Petricek. Technical dimensions of programming systems. *Art Sci. Eng. Program.*, 7(3), 2023. doi:10.22152/PROGRAMMING-JOURNAL.ORG/2023/7/13.
- 7 Tomas Petricek. Data exploration through dot-driven development. In Peter Müller, editor, *31st European Conference on Object-Oriented Programming, ECOOP 2017, June 19-23, 2017, Barcelona, Spain*, volume 74 of *LIPICs*, pages 21:1–21:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ECOOP.2017.21.
- 8 Tomas Petricek. Foundations of a live data exploration environment. *Art Sci. Eng. Program.*, 4(3):8, 2020. doi:10.22152/PROGRAMMING-JOURNAL.ORG/2020/4/8.
- 9 Tomas Petricek. The gamma: Programmatic data exploration for non-programmers. In Paolo Bottoni, Gennaro Costagliola, Michelle Brachman, and Mark Minas, editors, *2022 IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2022, Rome, Italy, September 12-16, 2022*, pages 1–7. IEEE, 2022. doi:10.1109/VL/HCC53370.2022.9833134.
- 10 Tomas Petricek, Gerrit J. J. van den Burg, Alfredo Nazábal, Taha Ceritli, Ernesto Jiménez-Ruiz, and Christopher K. I. Williams. AI assistants: A framework for semi-automated data wrangling. *IEEE Trans. Knowl. Data Eng.*, 35(9):9295–9306, 2023. doi:10.1109/TKDE.2022.3222538.
- 11 Don Syme, Keith Battocchi, Kenji Takeda, Donna Malayeri, and Tomas Petricek. Themes in information-rich functional programming for internet-scale data sources. In Evelyne Viegas, Karin K. Breitman, and Judith Bishop, editors, *Proceedings of the 2013 Workshop on Data Driven Functional Programming, DDFP 2013, Rome, Italy, January 22, 2013*, pages 1–4. ACM, 2013. doi:10.1145/2429376.2429378.
- 12 Jan Liam Verter and Tomas Petricek. Don't call us, we'll call you: Towards mixed-initiative interactive proof assistants for programming language theory. *CoRR*, abs/2409.13872, 2024. Presented at the 5th International Workshop on Human Aspects of Types and Reasoning Assistants (HATRA 2024). arXiv:2409.13872, doi:10.48550/ARXIV.2409.13872.

7 Typesetting instructions – Summary

LIPICs is a series of open access high-quality conference proceedings across all fields in informatics established in cooperation with Schloss Dagstuhl. In order to do justice to the high scientific quality of the conferences that publish their proceedings in the LIPICs series, which is ensured by the thorough review process of the respective events, we believe that LIPICs proceedings must have an attractive and consistent layout matching the standard of the series. Moreover, the quality of the metadata, the typesetting and the layout must also meet the requirements of other external parties such as indexing service, DOI registry, funding agencies, among others. The guidelines contained in this document serve as the baseline for the authors, editors, and the publisher to create documents that meet as many different requirements as possible.

Please comply with the following instructions when preparing your article for a LIPICs proceedings volume.

Minimum requirements

- Use pdf_latex and an up-to-date L^AT_EX system.
- Use further L^AT_EX packages and custom made macros carefully and only if required.
- Use the provided sectioning macros: `\section`, `\subsection`, `\subsubsection`, `\paragraph`, `\paragraph*`, and `\subparagraph*`.
- Provide suitable graphics of at least 300dpi (preferably in PDF format).
- Use BibTeX and keep the standard style (`plainurl`) for the bibliography.
- Please try to keep the warnings log as small as possible. Avoid overfull `\hboxes` and any kind of warnings/errors with the referenced BibTeX entries.
- Use a spellchecker to correct typos.

Mandatory metadata macros

Please set the values of the metadata macros carefully since the information parsed from these macros will be passed to publication servers, catalogues and search engines. Avoid placing macros inside the metadata macros. The following metadata macros/environments are mandatory:

- `\title` and, in case of long titles, `\titlerunning`.
- `\author`, one for each author, even if two or more authors have the same affiliation.
- `\authorrunning` and `\Copyright` (concatenated author names)
The `\author` macros and the `\Copyright` macro should contain full author names (especially with regard to the first name), while `\authorrunning` should contain abbreviated first names.
- `\ccsdesc` (ACM classification, see <https://www.acm.org/publications/class-2012>).
- `\keywords` (a comma-separated list of keywords).
- `\relatedversion` (if there is a related version, typically the “full version”); please make sure to provide a persistent URL, e. g., at arXiv.
- `\begin{abstract}... \end{abstract}`.

Please do not ...

Generally speaking, please do not override the `lipics-v2021`-style defaults. To be more specific, a short checklist also used by Dagstuhl Publishing during the final typesetting is given below. In case of **non-compliance** with these rules Dagstuhl Publishing will remove

the corresponding parts of L^AT_EX code and **replace it with the lipics-v2021 defaults**. In serious cases, we may reject the LaTeX-source and expect the corresponding author to revise the relevant parts.

- Do not use a different main font. (For example, the `times` package is forbidden.)
- Do not alter the spacing of the `lipics-v2021.cls` style file.
- Do not use `enumitem` and `paralist`. (The `enumerate` package is preloaded, so you can use `\begin{enumerate}[(a)]` or the like.)
- Do not use “self-made” sectioning commands (e.g., `\noindent{\bf My Paragraph}`).
- Do not hide large text blocks using comments or `\iffalse ... \fi` constructions.
- Do not use conditional structures to include/exclude content. Instead, please provide only the content that should be published – in one file – and nothing else.
- Do not wrap figures and tables with text. In particular, the package `wrapfig` is not supported.
- Do not change the bibliography style. In particular, do not use author-year citations. (The `natbib` package is not supported.)

This is only a summary containing the most relevant details. Please read the complete document “LIPICs: Instructions for Authors and the `lipics-v2021` Class” for all details and don’t hesitate to contact Dagstuhl Publishing (<mailto:publishing@dagstuhl.de>) in case of questions or comments: <http://drops.dagstuhl.de/styles/lipics-v2021/lipics-v2021-authors/lipics-v2021-authors-guidelines.pdf>

8 Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet, consectetur adipiscing elit [?]. Praesent convallis orci arcu, eu mollis dolor. Aliquam eleifend suscipit lacinia. Maecenas quam mi, porta ut lacinia sed, convallis ac dui. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse potenti. Donec eget odio et magna ullamcorper vehicula ut vitae libero. Maecenas lectus nulla, auctor nec varius ac, ultricies et turpis. Pellentesque id ante erat. In hac habitasse platea dictumst. Curabitur a scelerisque odio. Pellentesque elit risus, posuere quis elementum at, pellentesque ut diam. Quisque aliquam libero id mi imperdiet quis convallis turpis eleifend.

► **Lemma 1** (Lorem ipsum). *Vestibulum sodales dolor et dui cursus iaculis. Nullam ullamcorper purus vel turpis lobortis eu tempus lorem semper. Proin facilisis gravida rutrum. Etiam sed sollicitudin lorem. Proin pellentesque risus at elit hendrerit pharetra. Integer at turpis varius libero rhoncus fermentum vitae vitae metus.*

Proof. Cras purus lorem, pulvinar et fermentum sagittis, suscipit quis magna.

Just some paragraph within the proof. Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

▷ **Claim 2.** content...

Proof. content...

1. abc abc abc

◀

◀

■ **Listing 1** Useless code.

```
for i:=maxint to 0 do
begin
    j:=square(root(i));
end;
```

► **Corollary 3** (Curabitur pulvinar, [?]). *Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.*

► **Proposition 4.** *This is a proposition*

Proposition 4 and Proposition 4 . . .

8.1 Curabitur dictum felis id sapien

Curabitur dictum Corollary 3 felis id sapien Corollary 3 mollis ut venenatis tortor feugiat. Curabitur sed velit diam. Integer aliquam, nunc ac egestas lacinia, nibh est vehicula nibh, ac auctor velit tellus non arcu. Vestibulum lacinia ipsum vitae nisi ultrices eget gravida turpis laoreet. Duis rutrum dapibus ornare. Nulla vehicula vulputate iaculis. Proin a consequat neque. Donec ut rutrum urna. Morbi scelerisque turpis sed elit sagittis eu scelerisque quam condimentum. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aenean nec faucibus leo. Cras ut nisl odio, non tincidunt lorem. Integer purus ligula, venenatis et convallis lacinia, scelerisque at erat. Fusce risus libero, convallis at fermentum in, dignissim sed sem. Ut dapibus orci vitae nisl viverra nec adipiscing tortor condimentum [?]. Donec non suscipit lorem. Nam sit amet enim vitae nisl accumsan pretium.

8.2 Proin ac fermentum augue

Proin ac fermentum augue. Nullam bibendum enim sollicitudin tellus egestas lacinia euismod orci mollis. Nulla facilisi. Vivamus volutpat venenatis sapien, vitae feugiat arcu fringilla ac. Mauris sapien tortor, sagittis eget auctor at, vulputate pharetra magna. Sed congue, dui nec vulputate convallis, sem nunc adipiscing dui, vel venenatis mauris sem in dui. Praesent a pretium quam. Mauris non mauris sit amet eros rutrum aliquam id ut sapien. Nulla aliquet fringilla sagittis. Pellentesque eu metus posuere nunc tincidunt dignissim in tempor dolor. Nulla cursus aliquet enim. Cras sapien risus, accumsan eu cursus ut, commodo vel velit. Praesent aliquet consectetur ligula, vitae iaculis ligula interdum vel. Integer faucibus faucibus felis.

- Ut vitae diam augue.
- Integer lacus ante, pellentesque sed sollicitudin et, pulvinar adipiscing sem.
- Maecenas facilisis, leo quis tincidunt egestas, magna ipsum condimentum orci, vitae facilisis nibh turpis et elit.

► **Remark 5.** content...

9 Pellentesque quis tortor

Nec urna malesuada sollicitudin. Nulla facilisi. Vivamus aliquam tempus ligula eget ornare. Praesent eget magna ut turpis mattis cursus. Aliquam vel condimentum orci. Nunc congue,

libero in gravida convallis [?], orci nibh sodales quam, id egestas felis mi nec nisi. Suspendisse tincidunt, est ac vestibulum posuere, justo odio bibendum urna, rutrum bibendum dolor sem nec tellus.

► **Lemma 6** (Quisque blandit tempus nunc). *Sed interdum nisl pretium non. Mauris sodales consequat risus vel consectetur. Aliquam erat volutpat. Nunc sed sapien ligula. Proin faucibus sapien luctus nisl feugiat convallis faucibus elit cursus. Nunc vestibulum nunc ac massa pretium pharetra. Nulla facilisis turpis id augue venenatis blandit. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.*

Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.

10 Morbi eros magna

Morbi eros magna, vestibulum non posuere non, porta eu quam. Maecenas vitae orci risus, eget imperdiet mauris. Donec massa mauris, pellentesque vel lobortis eu, molestie ac turpis. Sed condimentum convallis dolor, a dignissim est ultrices eu. Donec consectetur volutpat eros, et ornare dui ultricies id. Vivamus eu augue eget dolor euismod ultrices et sit amet nisi. Vivamus malesuada leo ac leo ullamcorper tempor. Donec justo mi, tempor vitae aliquet non, faucibus eu lacus. Donec dictum gravida neque, non porta turpis imperdiet eget. Curabitur quis euismod ligula.

References

- 1 Michael D. Adams, Eric Griffis, Thomas J. Porter, Sundara Vishnu Satish, Eric Zhao, and Cyrus Omar. Grove: A bidirectionally typed collaborative structure editor calculus. *Proc. ACM Program. Lang.*, 9(POPL), January 2025. doi:10.1145/3704909.
- 2 Thorsten Altenkirch, Veronica Gaspes, Bengt Nordström, and Björn von Sydow. A user's guide to alf. Technical report, Chalmers University of Technology, Sweden, 1994. Unpublished Draft. URL: <https://people.cs.nott.ac.uk/psztxa/pub1/alf94.pdf>.
- 3 Edwin Brady. *The Idris Programming Language*, pages 115–186. Springer International Publishing, Cham, 2015. doi:10.1007/978-3-319-15940-9_4.
- 4 Damian Frölich and L. Thomas van Binsbergen. On the soundness of auto-completion services for dynamically typed languages. In *Proceedings of the 23rd ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences*, GPCE '24, page 107–120, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3689484.3690734.
- 5 Richard P. Gabriel. The structure of a programming language revolution. In *Proceedings of the ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, Onward! 2012, page 195–214, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2384592.2384611.
- 6 Joel Jakubovic, J. Edwards, and T. Petricek. Technical dimensions of programming systems. *Art Sci. Eng. Program.*, 7(3), 2023. doi:10.22152/PROGRAMMING-JOURNAL.ORG/2023/7/13.
- 7 Tomas Petricek. Data exploration through dot-driven development. In Peter Müller, editor, *31st European Conference on Object-Oriented Programming, ECOOP 2017, June 19-23, 2017, Barcelona, Spain*, volume 74 of *LIPICs*, pages 21:1–21:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ECOOP.2017.21.
- 8 Tomas Petricek. Foundations of a live data exploration environment. *Art Sci. Eng. Program.*, 4(3):8, 2020. doi:10.22152/PROGRAMMING-JOURNAL.ORG/2020/4/8.

- 9 Tomas Petricek. The gamma: Programmatic data exploration for non-programmers. In Paolo Bottoni, Gennaro Costagliola, Michelle Brachman, and Mark Minas, editors, *2022 IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2022, Rome, Italy, September 12-16, 2022*, pages 1–7. IEEE, 2022. doi:10.1109/VL/HCC53370.2022.9833134.
- 10 Tomas Petricek, Gerrit J. J. van den Burg, Alfredo Nazábal, Taha Ceritli, Ernesto Jiménez-Ruiz, and Christopher K. I. Williams. AI assistants: A framework for semi-automated data wrangling. *IEEE Trans. Knowl. Data Eng.*, 35(9):9295–9306, 2023. doi:10.1109/TKDE.2022.3222538.
- 11 Don Syme, Keith Battocchi, Kenji Takeda, Donna Malayeri, and Tomas Petricek. Themes in information-rich functional programming for internet-scale data sources. In Evelyne Viegas, Karin K. Breitman, and Judith Bishop, editors, *Proceedings of the 2013 Workshop on Data Driven Functional Programming, DDFP 2013, Rome, Italy, January 22, 2013*, pages 1–4. ACM, 2013. doi:10.1145/2429376.2429378.
- 12 Jan Liam Verter and Tomas Petricek. Don’t call us, we’ll call you: Towards mixed-initiative interactive proof assistants for programming language theory. *CoRR*, abs/2409.13872, 2024. Presented at the 5th International Workshop on Human Aspects of Types and Reasoning Assistants (HATRA 2024). arXiv:2409.13872, doi:10.48550/ARXIV.2409.13872.

A Styles of lists, enumerations, and descriptions

List of different predefined enumeration styles:

■ \begin{itemize}...\end{itemize}

■ ...

■ ...

1. \begin{enumerate}...\end{enumerate}

2. ...

3. ...

(a) \begin{alphaenumerate}...\end{alphaenumerate}

(b) ...

(c) ...

(i) \begin{romanenumerate}...\end{romanenumerate}

(ii) ...

(iii) ...

(1) \begin{bracketenumerate}...\end{bracketenumerate}

(2) ...

(3) ...

Description 1 \begin{description} \item[Description 1] ... \end{description}

Description 2 Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.

Description 3 ...

Proposition 10 and Proposition 10 ...

B Theorem-like environments

List of different predefined enumeration styles:

► **Theorem 7.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Lemma 8.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Corollary 9.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Proposition 10.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Conjecture 11.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Observation 12.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Exercise 13.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Definition 14.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Example 15.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Note 16.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Note.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Remark 17.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

► **Remark.** *Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.*

▷ **Claim 18.** Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.

▷ **Claim.** Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque.

Proof. Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque. ◀

Proof. Fusce eu leo nisi. Cras eget orci neque, eleifend dapibus felis. Duis et leo dui. Nam vulputate, velit et laoreet porttitor, quam arcu facilisis dui, sed malesuada risus massa sit amet neque. ◀

