调研论文：

1. **Static Analyses in Python Programming Courses**

* Students learning to program often rely on feedback from the compiler and from instructor-provided test cases to help them identify errors in their code.

学习编程的学生通常依赖于编译器和教师提供的测试用例的反馈，以帮助他们识别代码中的错误。

* This feedback focuses on functional correctness, and the output, which is often phrased in technical language, may be difficult to for novices to understand or effectively use.

这种反馈侧重于功能的正确性，而输出通常是用技术语言表达的，对于新手来说可能很难理解或有效地使用。

* Static analyses may be effective as a complementary aid, as they can highlight common errors that may be potential sources of problems.

静态分析可能是一种有效的辅助手段，因为它们可以突出可能是问题潜在来源的常见错误。

* In this paper, we introduce PyTA, a wrapper for pylint that provides custom checks for common novice errors as well as improved messages to help students fix the errors that are found.

在本文中，我们将介绍PyTA，这是一种pylint的包装器，它提供对常见的新错误的自定义检查，以及帮助学生修复发现的错误的改进消息。

* We report on our experience integrating PyTA into an existing online system used to deliver programming exercises to CS1 students and evaluate it by comparing exercise submissions collected from the integrated system to previously collected data.

我们报告了将PyTA集成到现有在线系统中的经验，该系统用于向CS1学生提供编程练习，并通过将从集成系统收集到的练习提交与以前收集到的数据进行比较来对其进行评估。

* This analysis demonstrates that, for students who chose to read the PyTA output, we observed a decrease in time to solve errors, occurrences of repeated errors, and submissions to complete a programming problem.

这个分析表明，对于选择阅读PyTA输出的学生，我们观察到解决错误、重复错误的出现和完成编程问题的提交的时间减少了。

* This suggests that PyTA, and static analyses in general, may help students identify functional issues in their code not highlighted by compiler feedback and that static analysis output may help students more quickly identify debug their code.

这表明，PyTA和一般的静态分析可以帮助学生识别编译器反馈没有突出显示的代码中的功能问题，而静态分析输出可以帮助学生更快地识别调试代码。

1. **Teaching object-oriented programming in python**

* Python's use in education has grown rapidly, due to its elegantly simple syntax.

Python在教育中的应用由于其优雅的简单语法而迅速增长。

* Though often viewed as a "scripting language," Python is a fully object-oriented language with an extremely consistent object model and a rich set of built-in classes.

尽管常常被看作是一种“脚本语言”，但Python是一种完全面向对象的语言，具有非常一致的对象模型和丰富的内置类集。

* In this tutorial, we share our experiences using Python in the context of an object-oriented CS1 course.

在本教程中，我们将在面向对象的CS1课程中分享使用Python的经验。

* We will begin with an overview of the language, with particular emphasis on the object-orientation.

我们将从语言的概述开始，特别强调面向对象。

* We then present several coherent teaching strategies and a variety of graphical and non-graphical projects.

然后我们提出几个连贯的教学策略和各种图形和非图形项目。

* Both new and experienced Python users are welcome.

欢迎新用户和有经验的Python用户。

1. **Fix the First, Ignore the Rest: Dealing with Multiple Compiler Error Messages**

* In order to help students learning to develop computer programs, several computing education researchers have analyzed the compiler error messages generated by novices' attempts to compile their programs.
* 为了帮助学生学习开发计算机程序，一些计算机教育研究人员分析了由新手尝试编译他们的程序所产生的编译错误信息。
* The goal is to help students diagnose the errors they make through the messages generated by the compiler.
* 其目的是帮助学生通过编译器生成的消息来诊断他们所犯的错误。
* This paper builds on that previous work by applying a technique based on a heuristic well-known to programmers - fix the first error and ignore the rest - to the analysis of over 21 million compiler error messages from the Blackbox dataset.
* 本文在前面工作的基础上，应用了一种技术，这种技术基于程序员所熟知的一种启发式方法——修正第一个错误，忽略其余错误——来分析来自Blackbox数据集的2100多万条编译器错误消息。
* We find that the ranks and frequencies obtained by considering all error messages are generally consistent with previously published lists, but when we consider first messages only, these ranks and frequencies are different.
* 我们发现，通过考虑所有错误消息得到的秩和频率通常与以前发布的列表一致，但是当我们只考虑第一个消息时，这些秩和频率是不同的。
* These differences could have important implications for teaching, and can inform tool design and future research efforts.
* 这些差异可能会对教学产生重要的影响，并为工具设计和未来的研究工作提供信息。

1. **Syntax-Based Analysis of Programming Concepts in Python**

* Writing programs is essential to learning programming.
* 编写程序是学习编程的必要条件。
* Most programming courses encourage students to practice with lab and homework assignments.
* 大多数编程课程都鼓励学生通过实验和家庭作业进行练习。
* By analyzing solutions to these exercises teachers can discover mistakes and concepts students are struggling with, and use that knowledge to improve the course.
* 通过分析这些练习的解决方案，教师可以发现学生正在挣扎的错误和概念，并利用这些知识来改进课程。
* Students however tend to submit many different programs even for simple exercises, making such analysis difficult.
* 然而，学生们倾向于提交许多不同的程序，甚至是简单的练习，这使得分析变得困难。
* We propose using tree regular expressions to encode common patterns in programs.
* 我们建议使用树正则表达式对程序中的常见模式进行编码。
* Based on these patterns we induce rules describing common approaches and mistakes for a given assignment.
* 基于这些模式，我们归纳出了描述给定作业的常见方法和错误的规则。
* In this paper we present a case study of rule-based analysis for an introductory Python exercise.
* 在本文中，我们将提供一个基于规则的分析的案例研究，作为Python入门练习。
* We show that our rules are easy to interpret, and can be learned from a relatively small set of programs.
* 我们表明，我们的规则很容易解释，并且可以从相对较小的一组程序中学习。

1. **The Error Behind The Message: Finding the Cause of Error Messages in Python**

* The interaction between a novice programmer, and the compiler plays a crucial role in the learning process of the novice programmer.
* 在初学者的学习过程中，编译器与初学者的相互作用起着至关重要的作用。
* Of particular importance is the compiler's feedback on errors in the program code.
* 特别重要的是编译器对程序代码中的错误的反馈。
* Accordingly, compiler error messages are an important and active field of research.
* 因此，编译器错误消息是一个重要而活跃的研究领域。
* Yet, a language that has largely been left out of this discussion so far is Python.
* 然而，到目前为止，有一种语言很大程度上被排除在这个讨论之外，那就是Python。
* We have collected Python programs from high school students taking introductory courses.
* 我们已经收集了Python程序从高中学生的入门课程。
* For each collected erroneous program, we sought to classify the effective error, and assess if the student was able to fix the error.
* 对于每个收集到的错误程序，我们试图对有效的错误进行分类，并评估学生是否能够修复错误。
* Our study is a precursor to providing improved error messages in Python, and assess their effectiveness.
* 我们的研究是在Python中提供改进的错误消息的先驱，并评估它们的有效性。
* As such, we are eventually interested in finding ways to automatically determine the effective error, so as to base the displayed message on.
* 因此，我们最终感兴趣的是寻找自动确定有效错误的方法，以便将显示的消息作为基础。
* From our data, we found that a considerable part of students' errors can be attributed to minor mistakes, which can easily be identified and corrected.
* 从我们的数据中，我们发现相当一部分学生的错误可以归因于小错误，这些小错误很容易识别和纠正。
* However, beyond such minor mistakes, a proper error diagnosis might have to be based on a goal/plan analysis of the entire program.
* 然而，除了这些小错误之外，正确的错误诊断可能必须基于整个程序的目标/计划分析。
* Likewise, proper assessment of whether an error has been fixed frequently requires more context than is provided by the program alone.
* 同样，对于是否修复了错误的正确评估常常需要比程序单独提供的更多的上下文。

1. **The Error Landscape: Characterizing the Mistakes of Novice Programmers**

* The software development process often follows a circuitous path, littered with mistakes and backtracks.
* 软件开发过程通常遵循一条迂回的道路，充满了错误和倒退。
* This is particularly true for novice programmers, who typically navigate through a variety of errors en route to their final solution.
* 对于新手来说尤其如此，他们通常会在各种各样的错误中摸索，最终找到解决方案。
* This paper presents a quantitative analysis of a large dataset of Python programs written by novice students.
* 本文对初学者编写的大型Python程序数据集进行了定量分析。
* The analysis paints a multifaceted picture of the errors that students encounter, providing insight into the distribution, duration, and evolution of these errors.
* 该分析描绘了学生们所遇到的错误的多面图，提供了对这些错误的分布、持续时间和演变的深入了解。
* Ultimately, this paper aims to incite further conversation on the mistakes made by novice programmers, and to inform the decisions instructors make as they help students overcome these mistakes.
* 最后，这篇论文的目的是激发更多关于新手程序员所犯错误的讨论，并告诉指导教师在帮助学生克服这些错误时所做的决策。

1. **Looking Glass: A C++ Library for Testing Student Programs through Reflection**

* Looking Glass (https://github.com/sturner84/LookingGlass) is a library designed to support the testing of students' C++ code through reflection.
* LookingGlass (https://github.com/sturner84/LookingGlass)是一个库，用来支持通过反射测试学生的c++代码。
* Instructors can design tests that compile even if there are mismatches between the instructor's and student's code (like misspelled function names or parameters that do not have the expected types).
* 教师可以设计可编译的测试，即使在教师和学生的代码之间存在不匹配(如拼写错误的函数名或不具有预期类型的参数)。
* Using the automatically generated metadata, instructors can search for and call functions or methods in the student code, including the main function, while providing feedback (rather than a compile error) if the required code is not found.
* 使用自动生成的元数据，教师可以搜索并调用学生代码中的函数或方法(包括主函数)，如果没有找到所需的代码，则提供反馈(而不是编译错误)。
* Although not suitable for all C++ programs, Looking Glass was designed to support instructors in introductory and intermediate classes and to be transparent to the students.
* 虽然不适合所有的c++程序，但是“镜子”的设计是为了支持入门和中级课程的教师，并且对学生是透明的。

1. **FrenchPress Gives Students Automated Feedback on Java Program Flaws**

* We created an Eclipse plug-in called FrenchPress that partially automates the task of giving students feedback on their Java programs.

我们创建了一个名为FrenchPress的Eclipse插件，它在一定程度上自动化了向学生提供Java程序反馈的任务。

* It is designed not for novices but for students taking their second or third Java course: students who know enough Java to write a working program but lack the judgment to recognize bad code when they see it.

它不是为初学者设计的，而是为正在学习第二或第三门Java课程的学生设计的:这些学生知道足够的Java来编写一个工作程序，但是当他们看到不好的代码时却缺乏识别能力。

* FrenchPress does not diagnose compile-time or runtime errors, or logical errors that produce incorrect output.

FrenchPress不诊断编译时或运行时错误，或产生不正确输出的逻辑错误。

* It targets silent flaws, flaws the student is unable to identify for himself because nothing in the programming environment alerts him.

它针对的是隐藏的缺陷，学生自己无法识别的缺陷，因为编程环境中没有任何东西能提醒他。

* FrenchPress diagnoses flaws characteristic of programmers who have not yet assimilated the object-oriented idiom.

FrenchPress诊断那些还没有理解面向对象习惯用法的程序员所特有的缺陷。

* Such shortcomings include misuse of the public modifier, fields that should have been local variables, and instance variables that should have been class constants.

这些缺点包括误用公共修饰符、本应是局部变量的字段和本应是类常量的实例变量。

* Other rules address the all too common misunderstanding of the boolean datatype.

其他规则解决了对布尔数据类型的常见误解。

* FrenchPress delivers explanatory messages in a vocabulary appropriate to the student's current level.

法语出版社用适合学生当前水平的词汇来传递解释性信息。

* This paper reports preliminary results of a formative evaluation of FrenchPress conducted in a Fall 2014 data structures and algorithms course.

本文报告了在2014年秋季数据结构与算法课程中对法国出版社进行的形成性评价的初步结果。

* User satisfaction survey responses indicate that among the students who received substantive diagnostic suggestions from FrenchPress, the percentage who were motivated to modify their program varied from 36% to 64% on four different assignments.

用户满意度调查结果显示，在从法国出版社获得实质性诊断建议的学生中，有动机修改程序的比例在四项不同的作业中从36%到64%不等。

1. **Online python tutor: embeddable web-based program visualization for cs education**

* This paper presents Online Python Tutor, a web-based program visualization tool for Python, which is becoming a popular language for teaching introductory CS courses.
* 本文介绍了一种基于web的Python程序可视化工具——Python在线导师，该工具正在成为一种流行的CS导论课程的教学语言。
* Using this tool, teachers and students can write Python programs directly in the web browser (without installing any plugins), step forwards and backwards through execution to view the run-time state of data structures, and share their program visualizations on the web.
* 使用这个工具，教师和学生可以直接在web浏览器中编写Python程序(不需要安装任何插件)，通过执行向前和向后执行来查看数据结构的运行时状态，并在web上共享他们的程序可视化。
* In the past three years, over 200,000 people have used Online Python Tutor to visualize their programs.
* 在过去的三年中，超过200,000人使用在线Python导师来可视化他们的程序。
* In addition, instructors in a dozen universities such as UC Berkeley, MIT, the University of Washington, and the University of Waterloo have used it in their CS1 courses.
* 此外，十多所大学，如加州大学伯克利分校、麻省理工学院、华盛顿大学和滑铁卢大学的教师也在他们的CS1课程中使用了它。
* Finally, Online Python Tutor visualizations have been embedded within three web-based digital Python textbook projects, which collectively attract around 16,000 viewers per month and are being used in at least 25 universities.
* 最后，在线Python导师可视化已经嵌入到三个基于web的数字Python教科书项目中，这三个项目每个月总共吸引了大约16000名读者，并且至少在25所大学中使用。
* Online Python Tutor is free and open source software, available at pythontutor.com.
* Python在线导师是免费的开源软件，可以在pythontutor.com上找到。

1. **Python predictive analysis for bug detection**

* Python is a popular dynamic language that allows quick software development.
* Python是一种流行的动态语言，允许快速的软件开发。
* However, Python program analysis engines are largely lacking.
* 然而，Python程序分析引擎在很大程度上是缺乏的。
* In this paper, we present a Python predictive analysis.
* 在本文中，我们提出了一个Python的预测分析。
* It first collects the trace of an execution, and then encodes the trace and unexecuted branches to symbolic constraints.
* 它首先收集执行的跟踪，然后将跟踪和未执行的分支编码为符号约束。
* Symbolic variables are introduced to denote input values, their dynamic types, and attribute sets, to reason about their variations.
* 符号变量被引入来表示输入值、它们的动态类型和属性集，以推断它们的变化。
* Solving the constraints identifies bugs and their triggering inputs.
* 解决约束可以识别缺陷及其触发输入。
* Our evaluation shows that the technique is highly effective in analyzing real-world complex programs with a lot of dynamic features and external library calls, due to its sophisticated encoding design based on traces.
* 我们的评估表明，该技术在分析具有大量动态特性和外部库调用的真实复杂程序时非常有效，这是因为它基于跟踪的复杂编码设计。
* It identifies 46 bugs from 11 real-world projects, with 16 new bugs.
* 它从11个实际项目中识别出46个bug，其中有16个是新bug。
* All reported bugs are true positives.
* 所有报告的bug都是真正的阳性。

1. **Teaching object-oriented programming in python**

* Python's use in education has grown rapidly, due to its elegantly simple syntax.

Python在教育中的应用由于其优雅的简单语法而迅速增长。

* Though often viewed as a "scripting language," Python is a fully object-oriented language with an extremely consistent object model and a rich set of built-in classes.

尽管常常被看作是一种“脚本语言”，但Python是一种完全面向对象的语言，具有非常一致的对象模型和丰富的内置类集。

* In this tutorial, we share our experiences using Python in the context of an object-oriented CS1 course.

在本教程中，我们将在面向对象的CS1课程中分享使用Python的经验。

* We will begin with an overview of the language, with particular emphasis on the object-orientation.

我们将从语言的概述开始，特别强调面向对象。

* We then present several coherent teaching strategies and a variety of graphical and non-graphical projects.

然后我们提出几个连贯的教学策略和各种图形和非图形项目。

* Both new and experienced Python users are welcome.

欢迎新用户和有经验的Python用户。

1. **Teaching Python programming with automatic assessment and feedback provision**

* We describe a method of automatic feedback provision for students learning programming and computational methods in Python.
* 介绍了一种面向学生学习Python编程和计算方法的自动反馈提供方法。
* We have implemented, used and refined this system since 2009 for growing student numbers, and summarise the design and experience of using it.
* 我们从2009年开始实施、使用和完善这个系统，以满足不断增长的学生人数，并总结了使用该系统的设计和经验。
* The core idea is to use a unit testing framework: the teacher creates a set of unit tests, and the student code is tested by running these tests.
* 核心思想是使用单元测试框架:教师创建一组单元测试，学生代码通过运行这些测试进行测试。
* With our implementation, students typically submit work for assessment, and receive feedback by email within a few minutes after submission.
* 在我们的实施中，学生通常会提交工作进行评估，并在提交后几分钟内通过电子邮件收到反馈。
* The choice of tests and the reporting back to the student is chosen to optimise the educational value for the students.
* 考试的选择和对学生的报告是为了优化学生的教育价值。
* The system very significantly reduces the staff time required to establish whether a student's solution is correct, and shifts the emphasis of computing laboratory student contact time from assessing correctness to providing guidance.
* 该系统大大减少了工作人员确定学生的解决方案是否正确所需的时间，并将计算实验室学生接触时间的重点从评估正确性转移到提供指导。
* The self-paced nature of the automatic feedback provision supports a student-centred learning approach.
* 自动反馈条款的自定进度特性支持以学生为中心的学习方法。
* Students can re-submit their work repeatedly and iteratively improve their solution, and enjoy using the system.
* 学生可以重复提交他们的工作，不断改进他们的解决方案，并享受使用系统。
* We include an evaluation of the system and data from using it in a class of 425 students.
* 我们对该系统进行了评估，并在一个425名学生的班级中使用了该系统的数据。