Types of Assignments for Novice Programmers

Alexander Ruf
Technische Universität München
TUM School of Education
Arcisstr. 21, D-80333 München
+49 89 289 29113
rufa@in.tum.de

Marc Berge
Technische Universität München
TUM School of Education
Arcisstr. 21, D-80333 München
+49 89 289 29112
berges@tum.de

Peter Hubwieser
Technische Universität München
TUM School of Education
Arcisstr. 21, D-80333 München
+49 89 289 29110
peter.hubwieser@tum.de

ABSTRACT
This paper deals with the classification of assignments according to their type. In contrast to other publications, we derive assignment types not deductively, but extract them empirically from different sources. Our main research question is: What types of programming assignments are actually given to novice programmers? In addition, we compare our empirically found assignment types to the deductively derived ones from the literature. This is driven by the following research questions: Are there types of assignments that are mentioned in literature, which however are not or rarely found in actual assignments given to novice programmers? Can assignment types be found that cannot or only poorly be matched with the category types described in the literature?

Categories and Subject Descriptors
K.3.2 [Computers and Education]: Computer and Information Science Education - Computer science education

Keywords
Types of Assignments, Task Types, Novice Programmers, Computer Science Education

1. RELATED WORK
In [1] Bower describes a taxonomy of task types in computing, deductively derived from curricula and literature and verified by experts. The types found there are one source for the comparison of our empirically derived types. The other source comes from Hazzan et al. [8] and Ragonis [10] respectively, where she published the question types from [8] for the olympiads in informatics. In both publications deductively derived questions are presented as a resource for teachers.

2. METHODOLOGY
The assignments to be analyzed should cover nearly all aspects concerning the population that is addressed by the assignments (students or self-learning readers). Also, different kinds of authors (e.g. teachers at school and university or authors of textbooks) should be covered as well. Therefore we investigated three groups of sources. The first group contains four Bavarian secondary school textbooks ([7], [3], [9], [4]). As second group we selected

the worksheets to be processed in the introductory course of our university, and as third group we chose two textbooks ([5], [6]) recommended in introductory programming courses in national and international universities (see [2]).

We included in our analysis all assignments of the chosen sources that contain programming code either in the assignment or in the corresponding solution. The extent of the programming code does not matter and ranges from just one line of code to the full program. Since we have restricted ourselves to assignments for novice programmers, we included assignments only up to the topics inheritance and polymorphism. Often, an assignment in the sources consists of several parts. Since the partial assignments usually differ in type we have treated and examined each subtask as an own assignment in these cases.

To identify the different types of assignments, we first looked at what is given in the respective assignment and what the student has to do to solve it. Then we stripped both criteria "given" and "to do" from the context of the assignment and formulated them in a generic way. Similar "givens" and "to dos" have been combined to one assignment type, i.e. two assignments are of the same type if they have basically the same given and if the same is to do. More complex assignments, which involve more than one "to do", were divided into corresponding parts and associated with multiple types, i.e. an "atomic" assignment was made from each to do, which was then used for further investigation. In a last step, we tried to derive a hierarchy within the found types.

3. THE RESULTING TYPES
Overall 967 tasks, fulfilling the criteria described above, were found in the seven sources referred to in section 2. From these 967 tasks 1098 assignments resulted. Within these 1098 assignments we could identify 14 different types. These were generalized as described above by searching for matches in what is given. We distinguished between what is given primarily and what may be given additionally. Then we categorized according to what is to do. As a result we got the following table of assignment types.

4. COMPARISON WITH DEDUCTIVELY DERIVED TYPES
If the task types listed by Bower in [1] are transferable on programming assignments, all of his types will be found in our empirically derived list. But the reverse is not the case, some of our types cannot be transferred to his, e.g. type 1.3 or type 2.1e. The reason for this may be because on the one hand the individual types in [1] are less accurately described and they are not specifically intended for programming assignments, on the other hand Bower’s objective was not a complete types list but a taxonomy within a list.
## 5. CONCLUSION AND FUTURE WORK

Overall, the results of the comparison indicate that our list is already quite complete, however, further investigations are needed to ensure completeness. More sources for assignments should be examined in the future, until we can assume to have found a saturated list of types of assignments. In particular, we have to study test assignments, since these have not been taken into account yet. Furthermore, the following aspects have not been investigated by us so far: Is it possible to conclude meaningfully from the type of an assignment to its difficulty? We don't think so, but one could at least argue that a certain level of taxonomy cannot be exceeded for some types. Are the different types of assignments motivating students differently or are students only motivated by the context of an assignment? To what extent is the context relevant to solve a (programming) assignment? The comparison of two assignments with different context implicitly assumes a classification of assignments. How can the various types of assignments be composed to more complex assignments? Is it possible to make general statements? In which form (e.g. multiple choice, free text etc.) can each type of assignment be presented? If there are several ways, will they all be found in practice? Are there any similar attempts to classify assignments in other subjects, such as in mathematics or physics? So, assignments are still a challenging field for research.

## 6. REFERENCES


### Table 1. The resulting assignment types

<table>
<thead>
<tr>
<th>Given</th>
<th>Additionally given</th>
<th>To do</th>
<th>Type No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nothing</td>
<td></td>
<td>Write a program (or a part of it)</td>
<td>1.1</td>
</tr>
<tr>
<td>Prerequisites</td>
<td></td>
<td>Write a program (or a part of it) considering the given prerequisites</td>
<td>1.2</td>
</tr>
<tr>
<td>Solution</td>
<td>Solution (=program code) to a similar problem or to a part of the problem</td>
<td>Adjust or extend the given solution to the problem</td>
<td>1.3</td>
</tr>
<tr>
<td>Solution</td>
<td>Solution (=program code) to the problem</td>
<td>Decide if the given solution is correct; give reasons for it or correct the solution</td>
<td>1.4a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set the right preconditions to the given solution</td>
<td>1.4b</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optimize the given solution</td>
<td>1.4c</td>
</tr>
<tr>
<td>Nothing</td>
<td>Transfer the given program code to your IDE and test it</td>
<td>2.1a</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Consider the effects of executing the given code</td>
<td>2.1b</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Draw a diagram to the given code</td>
<td>2.1c</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform the given code into a different programming language</td>
<td>2.1d</td>
<td></td>
</tr>
<tr>
<td>Prerequisites</td>
<td>Consider an appropriate problem to the given code</td>
<td>2.1e</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform the given code according to the given prerequisites</td>
<td>2.2</td>
<td></td>
</tr>
<tr>
<td>Diagram</td>
<td>Nothing</td>
<td>Write a program (or a part of it) to the given diagram</td>
<td>3</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>Nothing</td>
<td>Consider an assignment and solve it, considering the given prerequisites</td>
<td>4</td>
</tr>
</tbody>
</table>

The types list of Hazzan and Ragonis presented in [8] and [10], is much more extensive and more precisely described. From this list only two types cannot be integrated into our list: First, the type "completing a given solution" and second the type "efficiency estimation". That the latter is missing in our list is probably due to the fact that these assignments are made for more advanced and not for novice programmers, which we have studied. But it is in fact noteworthy that in none of our sources a "code cloze" occurs, especially since this type of assignment would be very suitable for beginners. Conversely, almost all of our assignment types can be transferred to the list of Hazzan and Ragonis. Of course, their classification differs in some points from ours, especially as their list is not only intended for programming assignments, nevertheless a correct mapping works almost always. Only type 2.1a, where program code is to be tested on the computer, does not match with Hazzan and Ragonis. This is probably because they have not considered this form of more practical work as a "typical" assignment.