# Rubrics for Student Outcome Assessment

**Outcome 1:** Analyze a complex computing problem and apply principles of computing and other relevant disciplines to identify solutions.

| | **Levels of Performance** | | | |
|---|---|---|---|---|
| **Indicators** | **Unsatisfactory** | **Developing** | **Satisfactory** | **Exemplary** |
| Abstraction of complex problems into formal models allowing for algorithmic solutions | Student is unable to model a complex problem formally | Student's attempts to model complex problems are inadequate | Student models complex problems sufficiently so that algorithmic solutions can be developed with effort | Student develops clean, elegant models of complex problems which easily translate into algorithmic solutions |
| Knowledge of adequate algorithmic strategies for solution development and implementation | Student is unable to select appropriate algorithmic strategies | Student is occasionally able to select appropriate algorithmic strategies | Student is often able to select appropriate algorithmic strategies | Student's algorithmic strategy selection is always appropriate and effective |
| Mapping formal specifications of problems into algorithmic solutions | Student exhibits little understanding of formal specifications | Student struggles at mapping formal specifications into algorithms | Student is usually able to map formal specifications into appropriate algorithms | Student is always able to map formal specifications into well-chosen algorithms |

**Rubric: one per student**

**Assessment method: direct**

**Assessment instruments: Select problems in exams, homework problems, programming assignments, and quizzes**

**Outcome 2:** Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline

| | Levels of Performance | | | |
|---|---|---|---|---|
| **Indicators** | **Unsatisfactory** | **Developing** | **Satisfactory** | **Exemplary** |
| Designs a programmable solution for a problem | Does not submit design | Submitted design is flawed and will not work | Submits workable design | Submits easily followed design using appropriate models and design methods |
| Implements an executable solution | Solution does not compile or run | Program compiles but does not run correctly | Program compiles and runs successfully | Solution is well-tested and appropriately documented |
| Validates the implementation relative to requirements | Does not attempt validation | Some requirements are validated, but a few requirements are not met | Meets all requirements | A rigorous implementation analysis is provided in relation to requirements, which are analyzed in detail |

**Rubric: one per student**

**Assessment method: direct**

**Assessment instruments: Programming assignments and programming projects**

**Example project: build a small kernel. This could be a group project, with components as individual projects.**

**Outcome 3:** Communicate effectively in a variety of professional contexts

| | Levels of Performance | | | |
|---|---|---|---|---|
| **Indicators** | **Unsatisfactory** | **Developing** | **Satisfactory** | **Exemplary** |
| Demonstrates effective written communication skills in the context of discussion topics | Shows little ability to write about discussion topics | Shows some ability to write about of discussion topics | Shows ability to write about discussion topics | Demonstrates mastery of ability to write about discussion topics |
| Effectively presents concepts in oral presentation | Content needs work both in research and organization, lacks effective visual aids, or is poorly timed | Content is understandable but could benefit from better research and organization, visual aids are ineffective, or is poorly timed | Content and research are appropriate and organized, visuals are adequate, and presentation timing was reasonable | Content is well-researched and organized, with effective visuals and well-planned presentation timing |

**Rubric: one per student**

**Assessment method: direct**

**Assessment instruments: discussion forum participation, oral presentations and term paper**

**Outcome 4:** Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles

| | **Levels of Performance** | | | |
|---|---|---|---|---|
| **Indicators** | **Unsatisfactory** | **Developing** | **Satisfactory** | **Exemplary** |
| Demonstrates knowledge of ethical, legal and social implications of computing | Demonstrates poor understanding of security, needs help to recognize basic elements of ethics | Demonstrates limited understanding of security, shows ability to recognize basic elements of ethics | Demonstrates good understanding of security, recognizes some elements of ethics | Demonstrates good understanding of security, recognizes basic elements of ethics |
| Demonstrates understanding of the impact of computing practices in society at large | Little understanding of the impact of computing practices in society | Some understanding of the impact of computing practices | Understands the impact of computing practices | Excellent ability to understand best computing practices in society |
| Demonstrates awareness of the significance of privacy and security in data management | Needs help | Demonstrates some ability | Demonstrates good ability | Demonstrates excellent ability to recognize significance of privacy and security in data management |

**Rubric: one per student**

**Assessment method: direct**

**Assessment instruments: Exams, homework's, term paper**

**Outcome 5:** Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline

| | Levels of Performance | | | |
|---|---|---|---|---|
| **Indicators** | **Unsatisfactory** | **Developing** | **Satisfactory** | **Exemplary** |
| Attendance and participation in team meetings | Does not attend or is late consistently | Does not attend all meetings, or attends meetings but meets few responsibilities | Attends all meetings, participates in all team activities and meets all responsibilities | Attends all meetings, contributes to discussions, and provides valuable initiative to team activities. |
| Contributes to overall group work product | Fails to contribute meaningfully to group work product | Contribution to group work product is occasionally helpful | Individual work consistently supports group effort | Individual work plays a critical role in group work product |
| Demonstrates leadership and teamwork skills | Does not communicate well | Sometimes participates in supporting group efforts and activities | Often takes initiative to support group efforts and activities | Always takes initiative, supports group efforts and encourages group members |

**Rubric: one per student**

**Assessment method: direct**

**Assessment instruments: team projects**

**Outcome 6:** Apply computer science theory and software development fundamentals to produce computing-based solutions

| | **Levels of Performance** | | | |
|---|---|---|---|---|
| **Indicators** | **Unsatisfactory** | **Developing** | **Satisfactory** | **Exemplary** |
| Working knowledge of programming languages and software development fundamentals | Student's command of even a single programming language is uneven | Student can function within a single programming language, but often fails to develop maintainable, abstract code | Student is able to function in several programming paradigms, and usually develops maintainable, abstract code | Student selects appropriate programming languages and paradigms for the task at hand, and always produces maintainable, abstract code |
| Knowledge of fundamental data structures and algorithms | Student does not understand fundamental data structures and algorithms | Student struggles at translating fundamental data structures and algorithms into appropriate code | Student usually produces workable implementations of fundamental data structures and algorithms | Student implements elegant implementations of fundamental data structures and algorithms |
| Knowledge of asymptotic notions and notations, and best-, expected- and worst-case analysis of algorithms | Student is not able to employ asymptotic analysis | Student understands the uses of asymptotic analysis, but rarely performs it correctly | Student usually performs asymptotic analysis appropriately and correctly | Student always performs asymptotic analysis appropriately and correctly |

**Rubric: one per student**

**Assessment method: direct**

**Assessment instruments: Select exam questions, homework, programming assignments, and quizzes**