Curriculum, Autumn 2017*

Bachelor in IT – Game Programming
Faculty of Technology

All information revised as of October 2016. Note that all curricula may be updated*
1 Introduction – Game Programming

The computer game industry is now a large international industry with programmers as some of its best-educated employees. Still, the programmers do not usually have a tailored education for game production. At the same time, more and more traditional industries use games and game-related technology.

Professional games are made by groups of varying size, but some roles will always be present: the game programmer develops new functionality and systems in games. Tasks that the programmer is typically responsible for are improved graphic effects, better artificial intelligence, and a more realistic physique. Another important activity for game programmers is tools used by other developers to create game content. With a focus on real time graphic simulations game programmers have good knowledge of the theories and technologies used.

In the Game Programming study the students will learn to realise exciting and realistic game experiences. They get a thorough introduction to C++ and other programming languages used in professional game development, and they learn to integrate existing technologies into new games.

1.1 Curriculum overview

The bachelor’s programme in game programming educates programmers who know how to develop games as well as heavy mathematics or graphics programs.

Bachelor in IT – Game Programming is a 3-year programme awarding the degree Bachelor in IT. The structure of the curriculum is shown in the matrix below. Courses offered to exchange students are described in Part 2.
Electives in the 2nd semester (Spring)
DS2100 Animation (7.5 ECTS)
DS2200 Digital Culture (7.5 ECTS)
PG2201 Unity Development (7.5 ECTS)

Electives in the 5th semester (Autumn) Bachelor in Game Programming
PG5500 Embedded Systems (7.5 ECTS)
PRO300 Virtual Reality Project (7.5 ECTS)
ENT200 Introduction to Entrepreneurship (7.5 ECTS)

The first year of study is jointly taught and provides students with solid basic qualifications in programming, project work, system development, data technique, and databases.

The second year gives a thorough introduction to the programming language C++, a much-used language for the development of large, modern games. C++ also provides a supplement to Java, which the students learn in the first year of study; thus, they master two widely used programming languages. The precursor to C++ is still widely used and the students therefore learn C as an introduction to C++. Further, they learn mathematics and physics with an emphasis on topics that are relevant for game and graphic programming. As a specialisation in the field of games they learn Game AI (artificial intelligence with a focus on game use). Since programming involves more than just programming languages, the students are also introduced to software architecture, giving them better understanding of the issues, as well as algorithms and data structures that will enhance their ability to write performance efficient code.

In the third year the students work with 3D graphics and its mathematical foundation. The tools used are C++ and the 3D system OpenGL. There are also electives in the third year and students can choose courses providing expertise for the game industry, or general knowledge of programming. The last semester includes the main project, the bachelor’s thesis, where the students solve a challenging task related to games, or to game or graphic programming, in cooperation with a private or public business enterprise. Additionally, there is a jointly taught core course in the Bachelor in IT: Research Methods.

1.2 About the programme
The bachelor’s programme in IT specialising in game programming aims to educate candidates with the proficiency to develop new functionality and systems in a game, which presupposes good competence in real time graphic simulations and wide knowledge of theories and technologies used to program games. An emphasis on practical work in cooperation with the industry, working in cross-disciplinary teams, provides the students with essential experience in dealing with complex issues, and a broad basis for success in their future careers in game development, improving graphic
effects, improving artificial intelligence and creating more realism in the graphic expression, as well as developing tools for content developers.

At completion of the programme candidates will be able to fulfil a wide spectre of programming roles in Norwegian and international business and industry. They will be of great interest to the game industry and also as program developers in C++ and Java, especially in environments where visual effects and multimedia are used as well as time-critical applications. The candidates will be characterised by a learning outcome defined by the following knowledge, skills and general competence:

Knowledge – candidates
• have wide knowledge of programming in general and especially in the languages Java, C# and C++, all of them relevant for game programming; they have good knowledge of program architecture, graphics programming, algorithms and data structures, and the use of mathematics, physics and artificial intelligence for games and simulators
• are familiar with research and development work in game and multimedia programming, and are able to update their knowledge in this field

Skills – candidates
• are able to use their programming knowledge in a wide spectre of activities, from solving general programming challenges in object oriented and program architecture, to more specialised challenges in artificial intelligence, simulation and graphics programming
• have wide experience in software tools, such as the versatile and well-known programming tool Visual Studio, and in more theoretical tools such as mathematics, for instance the use of quaternions in 3D graphics

General competence – candidates
• are able to take on work tasks and programming roles in the Norwegian and international business sector, thus being of great interest to the game industry and also to software developers in C#, C/C++ and Java, especially in domains where simulation, visual effects and/or artificial intelligence are used
• are able to estimate, plan and implement larger programming tasks, individually as well as in the capacity of members of agile teams (scrum), using version control programs as well as project management tools for this process.

1.3 Central themes
Bachelor in IT – Game Programming has the following focus:
Apart from regular programming knowledge such as software engineering and algorithms or data structures adapted to the field, knowledge of more specialised professional areas and research fields is required, such as linear algebra and special geometrical techniques, numerical approaches to classical mechanics, optimisation of graphic software, and artificial intelligence.
2 Individual courses offered to exchange students Autumn 2017

2.1 PG3300 Software Design

Norwegian name: Software Design
ECTS credits: 75
Area of study: Technology/IT
Language of instruction: English
Programme: Mandatory for Bachelor in IT – Programming and Bachelor in IT – Game Programming; optional course in Bachelor in IT – Intelligent Systems
Required prerequisites: Experience with basic object oriented programming, such as PGR100 Programming 1 and PGR101 Programming 2, or equivalent knowledge
Recommended prerequisites: None
Semester: The course is taught in the 3rd or 5th semester (Autumn)
Course leader: Tomas Sandnes

Course outline
The course will make students able to design and further develop extensive software systems using well-known techniques for modelling, testing and implementation.

Learning outcome
Knowledge: At completion of the course candidates will

- know the background and content for the UML standard
- know what Unit Testing is
- be able to explain the principles of test-driven development (TDD)
- know par programming and how the use of it influences software projects
- know what design patterns are
- know and be able to identify some important design patterns when reading them in code
- know what refactoring is
- know what multi-threading is
- understand how locking can be used to code thread securely
- be able to explain the principles of event handling

Skills: At completion of the course candidates will be able to

- know and use the basic syntax in the programming language C#, and know in what ways it is different from Java
- handle the UML diagrams: use case, class diagram, and sequence diagram
- use UML to design program architecture
- participate productively in par programming
- implement the following patterns: MVC & MVP, singleton, factory, builder, flyweight, composition, decorator
• use design guidelines such as layers and the GRASP principles emphasising the following for the latter: controller, information, expert, low coupling, high cohesion
• implement applications that apply several threads
• master an integrated development tool (IDE)
• write and edit source code with the mentioned tool

General competence: At completion of the course candidates will be able to
• reflect on multi threading and its usage
• cooperate with other programmers to develop good programs and to further develop their competence
• assess the quality of existing programs and possible structural improvements

Teaching and learning methods
Lectures, exercises, self-study

Recommended workload
Participation in lectures and tutorials – 48 hours
Self-study – 104 hours
Examination and preparing for the examination – 48 hours
Total recommended workload – 200 hours

Technology and tools
IDE: Visual Studio

Learning material/Syllabus
Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

Coursework requirements
None

Assessment
Assessment is based on a two-part examination consisting of the following elements:

• A written examination in groups (40%) for which the candidates submit self-developed program code and an accompanying document. The assignment has a time span of 3 weeks.
• An individual oral examination (60%) of 20 minutes. The oral examination takes its point of departure in the above-mentioned work (the written examination), and is used to decide an individual final grade adjusted according to the individual candidate’s performance as shown in the oral examination situation.

Assessment criteria
See Learning outcome
2.2 PRO300 Virtual Reality Project

Norwegian name: Virtual Reality-prosjekt

ECTS credits: 7.5

Area of study: Technology/IT

Language of instruction: English

Programme: Optional course in Bachelor in IT – Programming; Bachelor in IT – Game Programming; Bachelor in IT – Game Design; Bachelor in 3D Graphics; Bachelor in IT – Interactive Design; Bachelor in IT – Intelligent Systems

Required prerequisites: Basic qualifications in programming, interaction, game or concept development, or 3D design

Recommended prerequisites: As above

Semester: The course is taught in the 5th semester (Autumn)

Course leader: Kim Baumann Larsen

Course outline
The Virtual Reality Project is a cross-disciplinary course where students participate, bringing with them their different backgrounds. The purpose of the project is to develop interactive virtual reality applications for games, art or visualisation. The students will make concepts, design and create interactive Virtual Reality solutions, and thereby understand the technological frames in use.

Learning outcome
Knowledge: At completion of the course candidates will
- know the machine and software platforms for Virtual Reality
- know the platform requirements for different Virtual Reality solutions
- know the possibilities and limitations with Virtual Reality
- know the demands for achieving the largest possible presence in a Virtual Reality solution
- know basic principles for storytelling for interactive Virtual Reality

Skills: At completion of the course candidates will be able to
- draft and plan one interactive Virtual Reality concept for games, art or visualisation
- make prototypes for different platforms
- design and implement one interactive Virtual Reality experience for a chosen platform
- integrate 3D models, sound elements and interaction points into a holistic Virtual Reality experience for a chosen platform

General competence: At completion of the course candidates will be able to
- assess and create an optimal Virtual Reality solution for a chosen platform
- assess different Virtual Reality platforms and the possibility and limitations of solutions

Teaching and learning methods
Lectures and exercises, and one cross-disciplinary project in which students participate with their different study programme backgrounds either in programming, interaction, games/concept development, or 3D design.
**Recommended workload**
Participation in lectures and tutorials – 25 hours
Self-study – 80 hours
Independent preparation for presentations or discussions in class – 5 hours
Student work with projects, productions, assignments etc. – 20 hours
Independent exercises, lab work, practical work individually or in groups – 70 hours
**Total recommended workload – 200 hours**

**Technology and tools**
HTML editor

**Learning material/Syllabus**
Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

**Coursework requirements**
None

**Assessment**
Assessment is based on a combined examination consisting of the following elements:
- A group project examination (60%) where the groups submit the results of the project they have been working with in the course
- A group written examination (20%) consisting of a group report describing how the project was carried out, as well as reflections in connection with the work (a process document). The report should have a length of 3000-5000 words. The specifications of requirements for the report are handed out three weeks before submission.
- A group oral examination (20%) in which the groups present the project result and the process. The oral examination lasts approx. 25 minutes.

Grading scale: A-F with A as the best grade and E as the lowest pass grade. F means failed.
2.3 PG6200 Graphics Programming

Norwegian name: Grafikkprogrammering

ECTS credits: 7.5

Area of study: Technology/IT

Language of instruction: English

Programme: Mandatory course in Bachelor in IT – Game Programming

Required prerequisites: PG4400 C++ Programming, RF3100 Mathematics and Physics, or equivalent previous knowledge

Recommended prerequisites: None

Semester: The course is taught in the 5th semester (Autumn)

Course leader: Martin Lilleeng Sætra

Course outline
The course provides students with good overview of programming with C++ and modern OpenGL, and the central techniques and methods used. Further, the course teaches advanced methods of graphics and rendering (multi-pass algorithms, advanced lighting and shadows, etc.), using OpenGL and OpenGL Shading Language. The students also learn effective representation and treatment of geometry.

Learning outcome
Knowledge: At completion of the course candidates will know
- representation of geometry
- colours, light properties and material properties
- buffer objects
- frame buffer objects
- multi-pass algorithms
- per-pixel lighting
- blending
- texturation
- shadows
- cube maps
- normal mapping
- sub division

Skills: At completion of the course candidates will be able to
- work with geometrical transformations
- use quaternions, vectors and matrices for graphics programming
- work with various projections for graphics programming

General competence: At completion of the course candidates will
- possess good overview of OpenGL and OpenGL state machine
- have wide knowledge of 3D graphics programming, the use of shaders, and advanced topics of graphics and rendering
• know the main features of the development of OpenGL, and the development of graphics software and modern graphics cards architecture
• be comfortable with the programming of OpenGL in Visual Studio with C++
• have practical experience with the way OpenGL functions through modifications and extension of the given skeleton code

Teaching and learning methods
Lectures and exercises

Recommended workload
Participation in teaching – 24 hours
Self-study – 100 hours
Exercises – 24 hours
Examination and preparing for the examination – 52 hours
Total recommended workload – 200 hours

Technology and tools
Programming tools C++, OpenGL and OpenGL Shading Language

Learning material/Syllabus
Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

Coursework requirements
The prerequisite for taking the examination is that two mandatory assignments have been submitted and approved.

Assessment
Assessment is based on a three-part examination as described below:

• An individual written home examination (40%) lasting 2 weeks
• An individual written home examination (20%) lasting 2 weeks
• A final, individual oral examination (40%) lasting 20 minutes

Grading scale: A-F with A as the best grade and E as the lowest pass grade. F means failed.

Assessment criteria
The examination result is assessed in accordance with the expected learning outcome. The three parts of the examination are used to assess different aspects of the learning outcome.
2.4 PG5200 Tools Programming

Norwegian name: Tools programmering

ECTS credits: 7.5

Area of study: Technology/IT

Language of instruction: English

Programme: Mandatory course in Bachelor in IT – Game Programming and Bachelor in IT – Programming

Required prerequisites: PG2100 Programming 2, or equivalent previous knowledge

Recommended prerequisites: Experience with basic C# programming (such as the C# part of PG3300 Software Architecture, or the equivalent knowledge)

Semester: The course is taught in the 5th semester (Autumn)

Course leader: Kjetil Raaen

Course outline

The course provides the students with knowledge and understanding of the challenges and solutions when developing tools that enable people from other professions to produce content for games. The students develop a simple level editor and other tools that are needed to develop games and other multimedia products. Further, the students will be introduced to the need for stability, error handling and fast reaction.

Learning outcome

Knowledge: At completion of the course candidates will
- understand basic functionality in a game engine, and what the concept of tool chain means
- know various types of game engines and the difference between pure graphics engines and complete game engines
- understand why good tools are essential for effective production of content
- be able to explain the different categories of tools used in game development, including level editors, property editors, and support tools
- be able to describe the most important functionality in a level editor
- know third party solutions for game tools
- know network protocols and how games communicate over networks

Skills: At completion of the course candidates will be able to
- evaluate the benefit of making development tools oneself as opposed to using third party solutions
- develop specifications of requirement for a selection of tools
- develop tools in C# with WPF
- implement essential functionality for a level editor
- master the use and handling of exceptions for improved stability
- avoid heavy operations in GUI thread
- add support for scripting in existing code
- save game state and log events in a database
- serialise and de-serialise game data

General competence: At completion of the course candidates will
- understand how a good tool chain can improve development efficacy
- be able to reflect critically on the game development process and necessary demands

Teaching and learning methods
The course is taught in the form of 12 lectures of about 2 hours each and 12 teacher-led exercises of 2 hours each. The teacher-led exercises are not mandatory but to reach the competence goals the students are expected to complete the exercises on their own if the scheduled exercises are not sufficient. Additionally, the students have to focus on the home examination.

Recommended workload
Participation in lectures and tutorials – 24 hours
Self-study – 90 hours
Independent exercises, lab work, practical work individually or in groups – 24 hours
Examination and preparing for the examination – 62 hours
Total recommended workload – 200 hours

Technology and tools
Visual Studio IDE

Learning material/Syllabus
Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

Coursework requirements
None

Assessment
Assessment is based on an individual written home examination lasting 4 weeks.

Grading scale: A-F with A as the best grade and E as the lowest pass grade. F means failed.

Assessment criteria
The candidates are assessed by their ability to develop usable tools that can be integrated in a tool chain for the development of games and similar applications.
2.4 RF5100 Linear Algebra

Norwegian name: Lineær algebra

ECTS credits: 7.5

Area of study: Technology/IT

Language of instruction: English

Programme: Mandatory course in Bachelor in IT – Game Programming

Required prerequisites: Successfully completed PGR100 Object Oriented Programming 1, PGR101 Object Oriented Programming, and RF3100 Mathematics and Physics, or equivalent previous knowledge

Recommended prerequisites: None

Semester: The course is taught in the 5th semester (Autumn)

Course leader: Lars Sydnes

Course outline

In this course, the students are prepared for future jobs in three-dimensional graphics programming in that they are provided with the necessary knowledge of geometry, transformation theory, and linear algebra. Additionally, the course aims to give them general practice in the use of mathematical methods.

Learning outcome

Knowledge: At completion of the course candidates will

- know how to represent various geometrical objects by means of vectors
- know the use of linear and affine transformations, especially geometrical transformations such as translation and rotation
- know descriptions of rotation and space orientation, such as rotation matrices, Euler angles, rotation vectors, axis-angle representation, and quaternions
- know methods for converting between the mentioned descriptions of rotation and space orientation
- know homogenous vectors and matrices, especially matrices for central projection
- know methods for solving linear equation systems
- know the use of scalar product in connection with projections and approximations
- know barycentric coordinates and interpolation in triangles, and the use of this in 3D graphics, for instance in connection with Phong shading and texturing
- be acquainted with a number of applications for linear algebra, such as calculating interpolating curves

Skills: At completion of the course candidates will be able to

- solve geometrical problems by using the theory of vectors, matrices and linear equation systems in the areas described in the paragraph about knowledge outcome
- master combining and alternating between different coordinate systems such as model coordinates, world coordinates, camera coordinates, and screen coordinates
- apply linear algebra to a number of application areas
General competence: At completion of the course candidates will
  • have gained good overview of vector geometry and basic linear algebra with applications, in the areas described in the paragraph on knowledge outcome
  • have a good starting point for further work with graphics programming
  • have matured in their mathematical proficiency

Teaching and learning methods
Lectures and self-study with guidance from the lecturer

Recommended workload
Participation in lectures and tutorials – 48 hours
Self-study – 128 hours
Examination and preparing for the examination – 24 hours
Total recommended workload – 200 hours

Technology and tools
Paper and pencil, pc

Learning material/Syllabus
Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

Coursework requirements
None

Assessment
Assessment is based on a 3-hour individual written examination. Aids permitted: calculator

Grading scale: A – F, with A as the best grade and E as the lowest pass grade. F means failed.

Assessment criteria
See Learning outcome
2.5 PG5500 Embedded Systems

Norwegian name: Embedded Systems

ECTS credits: 7.5

Area of study: Technology/IT

Language of instruction: English

Programme: Mandatory course in Bachelor in IT – Intelligent Systems; optional course in other Bachelor in IT programmes

Required prerequisites: Good programming skills (corresponding to 3rd semester Bachelor in IT or equivalent knowledge)

Recommended prerequisites: None

Semester: The course is taught in the 4th semester as mandatory course in Intelligent Systems and in the 5th semester (Autumn) as elective for all Bachelor in IT programmes

Course leader: Tomas Sandnes

Course outline

The course provides students with knowledge of embedded systems and the use of embedded systems with architecture. The course also teaches basic hardware components and Arduino, and provides students with skills in programming applications that communicate with I/O units connected with micro controllers and Unix-based machines. The students will gain proficiency in areas of use and practical applications of embedded systems.

Learning outcome

Knowledge: At completion of the course candidates will
- understand the term “embedded systems” and know the difference between a CPU and a micro controller
- know the most used architectures for embedded systems
- know the hardware in Raspberry Pi and its use
- know Arduino and its use

Skills: At completion of the course candidates will be able to
- set up and install OS and other software on Raspberry Pi
- use basic electronic components such as LED, buttons, resistance, buzzer and transistors
- steer external components with analogue as well as digital signals, from an embedded controller (segment display, led matrix, LCD, I/O expansion)
- input data from analogue and digital external sensors (e.g. light, temperature, buttons)
- steer engines (servo, DC, stepper, solenoid, etc.)
- communicate wireless with IR
- use data from external sensors to steer external components
- use important protocols of embedded systems such as SPI, I2C, 1Wire, and UART
- understand how embedded systems function in the interface between electronics and program (eeprom, interrupts, memory handling)
• design simple circuit cards with Fritzing

General competence: At completion of the course candidates will
• know uses and practical applications of embedded systems
• design, test and critically evaluate embedded systems as solutions to real tasks, such as making robots and game consols

Teaching and learning methods
Lectures, exercises and self-study

Recommended workload
Participation in lectures and tutorials – 24 hours
Self-study – 71 hours
Independent preparation for presentations and discussions in class – 24 hours
Exercises – 10 hours
Examination and preparing for the examination – 200 hours

Technology and tools
• Arduino, Raspberry Pi and electronic components
• Fritzing

Learning material/Syllabus
Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

Coursework requirements
In order to take the examination candidates must have submitted 2 mandatory assignments with approval.

Assessment
Assessment is based on an individual project assignment lasting 2 weeks. The examination is about making a finished product based on the parts from the well-equipped component set used in the course.

Assessment criteria
The candidates’ performances are assessed by quality of idea, code, documentation, usability and creativity.
See also: Learning outcome
### 2.6 ENT200 Essentials of Entrepreneurship

**Norwegian name:** Innføring i entreprenørskap  

**ECTS credits:** 7.5  

**Area of study:** Management  

**Language of instruction:** English  

**Programme:** The course is mandatory in Bachelor in Project Management Arts and Creative Industries, and in Bachelor in Strategy and PR; optional course in Bachelor in IT – Game Programming  

**Requires prerequisites:** None  

**Recommended prerequisites:** None  

**Semester:** The course is taught in the 3rd semester as mandatory course, in the 5th semester as cross-faculty elective  

**Course leader:** Tor Grønsund  

---

#### Course outline

The course gives a theoretical introduction to and overview of the academic field of entrepreneurship set in a creative context. The students will learn how entrepreneurial opportunities may arise, be developed and realised. Central themes are the relationship between traditional and entrepreneurial management perspectives and entrepreneurs’ role in society.

---

#### Learning outcome

**Knowledge:** At completion of the course candidates will
- possess basic knowledge of central theories and research in the field of entrepreneurship  
- be able to account for the way entrepreneurial opportunities can arise and be developed  
- know the relationship between traditional and entrepreneurial management perspectives, and the differences between intrapreneurship and entrepreneurship  
- be able to reflect and critically evaluate points of view and issues related to entrepreneurship and the role of the entrepreneur in a creative context

**Skills:** At completion of the course candidates will be able to
- deal with the themes taught in the course in an independent way, in writing and orally  
- operationalise relevant theory in order to evaluate entrepreneurial processes and issues in a cultural and creative context

**General competence:** At completion of the course candidates will
- have the ability to communicate and present points of view and experiences with others who have a background and knowledge in entrepreneurial theory and practice, thus contributing to sound practice  
- understand and be able to contribute to discourses of entrepreneurship and the role of the entrepreneur in creative industries and society
Teaching and learning methods

Lectures, company visits, case assignments, group assignments, written assignments. Discussions based on the lectures and the required reading will give the students the possibility to learn actively, and it will strengthen their level of reflection. Written tasks and assignments with feedback as well as tutorials will strengthen their learning outcome.

Recommended workload

Participation in lectures and tutorials – 48 hours
Self-study – 88 hours
Independent preparation for presentations and discussions in class – 20 hours
Student work – projects, productions, assignments etc. – 20 hours
Independent exercises, lab work, practical work individually or in groups – 24 hours
Total recommended workload – 200 hours

Technology and tools

NA

Learning material/Syllabus

Updated information on required reading and other learning material is posted per programme on our electronic learning platform before the semester starts. The information is also available on our website.

In addition to literature and other learning material, scheduled teaching and other scheduled learning activities are part of the syllabus.

Coursework requirements

To be allowed to take the examination candidates must have participated in and passed an oral group presentation

Assessment

Assessment is based on an individual written home examination lasting 2 weeks. The examination paper should have a length of 8-10 A4 pages.

Grading scale: A – F with A as the best grade and E as the lowest pass grade. F means failed.