Text in black gives basic requirements. Text in blue gives advanced requirements. Every student at the beginning of the exam picks a favourite part out of the four below. Then a question is drawn uniformly and independently at random from each part. A student should answer the basic requirements in all questions and advanced in the question in the favourite topic.

In general, by a *sketch* of a proof we mean understanding statements of consecutive steps of the proof and a general idea how they are proved. A *full proof* means ability to reproduce a complete argument on the board, possibly with some slackness in numerical constants.

Part 1: Basics (Lectures 1, 2, 3)

1. Basic definitions: parameterized problem, XP, fixed-parameter tractability. Examples of different problems and parameterizations. 2^k * (n+m) FPT algorithm for Vertex Cover, and any improvement to c^k * (n+m) for some c<2. k^O(k) * (n+m) FPT algorithm for Feedback Vertex Set via branching (with proof). [Platypus 1 and 3.1-3.3]
2. Definition of kernelization. Proof that the existence of any kernelization algorithm is equivalent to the problem being FPT. Some easy examples of problems with polynomial kernels, including a quadratic kernel for Vertex Cover. Sunflower Lemma (statement or full proof) and its application for a O(k^d) kernel for d-Hitting Set. [Platypus 2.1, 2.2 and 2.6]
3. Crown Decomposition Lemma (statement and sketch or full proof) and its application to give a 3k kernel for Vertex Cover. [Platypus 2.3]
4. Iterative compression: presentation of the technique and its applications in two of the following problems: Vertex Cover, Feedback Vertex Set, Feedback Vertex Set in Tournaments, Odd Cycle Transversal. One of these examples should be the 5^k * poly(n) algorithm for Feedback Vertex Set. [Platypus 4, scanned notes]
5. 3^|T| * poly(n) algorithm for Steiner Tree and 2^|U| * poly(|U|+|F|) algorithm for Set Cover via dynamic programming on subsets. Two more examples of FPT algorithms obtained by dynamic programming on subsets. [Platypus 6.1]

Part 2: Color coding, algebraic techniques, repsets (Lectures 4, 7, 8, 11)

1. Color coding and random separation with two (three) different applications, including k-Path in time (2e)^k * poly(n) [Platypus 5.2.1], Subgraph Isomorphism in bounded degree graphs [Platypus 5.3] and long directed cycle in 4^k*poly(n), [Scanned notes, paper]. Statement of derandomization through perfect hash families and universal sets [Platypus 5.6].
2. Inclusion-Exclusion principle and at least one algorithmic application [Platypus 10.1]. Definitions of zeta and Moebius transforms, Inversion formula (statement), Yates' algorithm (computing all values of zeta/Mobius transform in 2^n*poly(n) time) [Platypus 10.2], fast cover product and fast subset convolution [Platypus 10.3], with applications to dynamic programming over tree decomposition [Platypus 11.1]. See also [scanned notes]
3. k-Path in 2^k*poly(n) time by a reduction to detecting monomials linear in a set of variables (proof that the monomial linear in x1..xk can be detected in a polynomial which

is k-homogeneous n x1..xk by a randomized algorithm in time 2^k) [scanned notes]. Remark: alternative exposition, by a direct 2^k algorithm for k-path [Platypus 10.4.1] is also permitted.

4. Definition of representative sets. Existence and computation of a small representative set using Gaussian elimination (statement or full proof) [Platypus 12.3.1]. One (two) examples of applications of parameterized algorithms (e.g. k-path in 5.19^k*poly(n) [scanned notes], kernels for d-Hitting Set, d-Set Packing [Platypus 12.3.2, 12.3.3])

Part 3: Treewidth, cut problems and applications of LP (Lectures 5, 6, 9, 10)

1. Definition of treewidth, nice tree decompositions, example of dynamic programming over a tree decomposition, Courcelle's theorem (statement), 4-approximation algorithm for treewidth in FPT time (statement or full proof). [Platypus 7.1-7.4 and 7.6]
2. Grid minor theorem and its applications for parameterized algorithms. Treewidth and grid minors in planar graphs. Bidimensionality: applications in exact and parameterized algorithms. Baker's technique in parameterized algorithms. Proof that the treewidth of a planar graph is linear in the radius. [Platypus 7.7]
3. Definition of important cuts. Upper bound on the number of important cuts of size at most k (statement and full proof). Application for a parameterized algorithm for Edge Multiway Cut in time 4^k * poly(n). [Platypus 8.1-8.3]
4. FPT algorithm for Directed Feedback Vertex Set (sketch or full proof) [Platypus 8.5 and 8.6]
5. Nemhauser-Trotter Theorem (proof) and its application in kernelization of vertex cover [Platypus 2.5]. LP-guided branching: 4^(k-LP)*poly(n) algorithm for vertex cover [Platypus 3.4]. Also see [scanned notes]

Part 4: Lower bounds (Lectures 12, 13, 15)

1. Basic definitions: FPT reductions, W[1]- and W[2]-hardness and completeness. Examples of W[1]- and W[2]-problems with corresponding reductions (including examples with gadgets for choosing edges). [Platypus 13.1, 13.2 and 13.6]
2. Definitions of ETH and SETH. Transferring ETH/SETH lower bounds via reductions. Examples, including non-linear parameter blow-ups. Statement of the sparsification Lemma, refutation of subexponential algorithm for 3SAT in terms of formula size (statement or full proof). Proof that SETH implies ETH. [Platypus 14.1, 14.2, 14.3.1]
3. Lower bounds for W-hard problems under ETH. Hardness of Clique under ETH (statement and sketch, or full proof). [Platypus 14.4]
4. Hardness based on SETH: k-DominatingSet: n^(k-epsilon), Independent Set for graphs with given path decomposition of width p: 1.99^p*poly(n), 2-Orthogonal Vectors: n^1.99, 1.4999-approximation of Diameter in n^1.999 time for sparse graphs (E=O(V)). Basic requirenents: all formulations, at least one reductions; advanced: at least 2 reductions, including Independent Set. [Platypus 14.5 + scanned notes]