RUSSIAN MINISTRY OF SCIENCE AND EDUCATION

FEDERAL STATE BUDGETARY EDUCATIONAL INSTITUTION
OF HIGHER EDUCATION
«BASHKIR STATE UNIVERSITY»

FACULTY OF MATHEMATICS AND INFORMATION TECHNOLOGIES

Approved: at the department meeting
Protocol # 5 from January 26, 2021
Head of the department

_____Yulmukhametov R.S.

Coordinated with:
EMC chairman of the faculty/institute

_____ Efimov A.M.

WORKING PROGRAM OF DISCIPLINE (MODULE)

Discipline Languages and methods of programming

(name of the discipline)

Obligatory part

(name of the part enclosing the discipline (obligatory, formed by participants of the educational activity, facultative))

bachelor (undergraduate) program

Course of training (speciality)
01.03.02 Applied mathematics and informatics
(code and name of the course of training (speciality))

Subdivision of the course of training (profile)
Applied programming and data analysis
(name of the profile of training)

Qualification (level of training)
bachelor
(name of the level of training)

Designer (compiler): associate professor of the PEI department, PhD	_____ Garifullin R.N.

For enrollment of: 2021

Ufa 2021

Designer: associate professor, PhD Garifullin Rustem Nailevich.

The working program of the discipline is approved at the meeting of the department of Programming and Economic Informatics,
protocol # 5/1 from January , « 26 » 2021.

Head of the department _____ Yulmukhametov R. S.

The addenda and updates introduced into the working program of the discipline are approved at the meeting of the department of Programming and Economic Informatics,,
protocol # 11 from June , « 15 » 2021.

Head of the department _____ Yulmukhametov R. S.

List of documents and materials

1. List of expected results of education in the discipline correlated with the indicators of reaching the competencies.
2. Goal and role of the discipline in the structure of educational program.
3. Content of the working program (duration of the discipline, sorts and forms of classes, educational and methodical support for the individual work of learners).
4. Fund of grading materials in the discipline.
4.1. List of competencies and indicators of reaching competencies with expected learning results in the discipline correlated with them. Description of criteria and scales for grading the learning results in the discipline.
4.2. Typical grading tasks or other materials required for grading the learning results in discipline correlated with the indicators of reaching the competencies which are set in the educational program. Methodical materials determining the grading procedures for learning results in the discipline.
5. Educational, methodical and informational support of the discipline.
5.1. List of references to primary and complementary educational literature necessary for acquiring the discipline.
5.2. List of the Internet resources and software necessary for acquiring the discipline, including professional data bases and reference systems.
6. Hardware equipment, materials and rooms necessary for implementing the educational process in the discipline.
Appendix 1
Appendix 2

1. List of intended learning outcomes of the discipline, correlated with the indicators of achievement of competencies set in the educational programme

The following learning outcomes should be achieved at the end of the discipline:

Competency being formed (with code)	Code and name of the indicator of reaching the competency	Learning results in the discipline
GPC-2 – Is able to use and adapt existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems.	GPC-2.1 Possesses fundamental knowledge of existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems.	Possesses fundamental knowledge of existing programming systems for the development and implementation of algorithms for solving applied problems.
	GPC-2.2. Knows how to use the existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems in professional activities.	Knows how to use the existing programming systems for the development and implementation of algorithms for solving applied problems.
	GP-2.3. Has the skills to apply existing mathematical methods and programming systems for the development and implementation of algorithms for solving specific problems.	Has the skills to use programming systems for the development and implementation of algorithms for solving specific problems.

2. Goal and role of the discipline in the structure of educational program.

The discipline (module) "Programming languages and methods" belongs to the basic part.

The discipline (module) is studied on courses 1-2 in 1,2,3,4 semesters.

The relevance of the discipline is due to the fact that informatics and its applications - information technology - penetrate all spheres of human activity. Therefore, learning basic concepts of languages and methods, basic algorithms of working with data is a prerequisite for the formation of a specialist in the field of computer science. This knowledge is needed in various fields to solve practical problems in a variety of application areas such as programming, mathematical processing and data transfer, pattern recognition, cryptography, etc.

The aims of the "Programming languages and methods" discipline are to learn the basic concepts of computer science and data structures, and basic algorithms for dealing with data. This knowledge is essential for solving practical problems in a variety of application areas, such as programming, mathematical processing and data transfer, pattern recognition, cryptography, etc.

3. Content of the working program (duration of the discipline, sorts and forms of classes, educational and methodical support for the individual work of learners).

The content of the work programme can be found in Appendix No. 1.

4. Fund of grading materials in the discipline.

4.1. List of competencies and indicators of reaching competencies with expected learning results in the discipline correlated with them. Description of criteria and scales for grading the learning results in the discipline.

The process of mastering the discipline (module) is aimed at forming the following competences:

GPC-2 – Is able to use and adapt existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems.

Code and name of the indicator of reaching the competency	Learning results in the discipline	Grading criteria for learning results	
		«Uncredited»	«Credited»
GPC-2.1 Possesses fundamental knowledge of existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems.	Possesses fundamental knowledge of existing programming systems for the development and implementation of algorithms for solving applied problems.	Lack of knowledge or fragmented knowledge of existing programming systems to develop and implement algorithms to	Generated (possibly incomplete) knowledge of existing programming systems to develop and implement algorithms to solve applied

		solve applied problems.	problems.
GPC-2.2. Knows how to use the existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems in professional activities.	Knows how to use the existing programming systems for the development and implementation of algorithms for solving applied problems.	Missing skills or fragmented skills to use the apparatus of of existing systems of programming systems to develop and implementation of algorithms for solving applied tasks.	An educated (perhaps unsystematic) ability to Use the apparatus of existing programming systems programming systems for the development and implementation of algorithms of solving applied problems.
GP-2.3. Has the skills to apply existing mathematical methods and programming systems for the development and implementation of algorithms for solving specific problems.	Has the skills to use programming systems for the development and implementation of algorithms for solving specific problems.	Missing or fragmented possession Fragmentary possession skills in applying the apparatus of programming systems for development and implementation algorithms in solving specific tasks.	Successful and systematic (possibly with minor deficiencies) mastery of of application of programming systems skills (with some minor gaps) programming systems for the development and implementation of algorithms for solving specific tasks.

Code and name of the indicator of reaching the competency	**Learning results in the discipline**	Grading criteria for learning results			
		«Unsatisfactory»	«Satisfactory»	«Good»	«Excellent»
GPC-2.1 Possesses fundamental knowledge of existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems.	Possesses fundamental knowledge of existing programming systems for the development and implementation of algorithms for solving applied problems.	Practically does not know	Has substantial gaps in the knowledge	Knows almost all	Knows all
GPC-2.2. Knows how to use the existing mathematical methods and programming systems for the development and implementation of algorithms for solving applied problems in professional activities.	Knows how to use the existing programming systems for the development and implementation of algorithms for solving applied problems.	Practically is unable	Is unable in most	Is able to do almost all	Is able to do all
GP-2.3. Has the skills to apply existing mathematical methods and programming systems for the	Has the skills to use programming systems for the development and implementation of algorithms for solving specific problems.	Practically does not have	Does not have in most	Has in essential	Has

development and implementation of algorithms for solving specific problems.					

The form of the final grading in the discipline in semesters 1,2 and 4 is an exam.

Criteria for grading are scores which are set by the instructor for various activities (grading tasks) in and upon learning modules listed in the Rating plan of the discipline.

from 45 up to 59 scores – «satisfactory»;

from 60 up to 79 scores – «good»;

from 80 scores up – «excellent».

The form of the final grading in the discipline in semesters 1,2 and 4 is an credit:

credit - 60 to 110 rating points (including 10 incentive points),

uncredited - from 0 to 59 rating points.

4.2.Model control tasks or other materials necessary to assess the knowledge, skills and activityexperience that characterise the stagesof competence formation in the process of mastering the educational programme. Methodological materials defining the evaluation procedures for knowledge, skills and activity experience, characterising the stages of competence formation

4.3 Rating plan for the discipline

The discipline rating plan can be found in the appendix 2.

Examination tickets

The examination (credit) is an assessment tool for all stages of competence acquisition.

Structure of the examination ticket: 2 questions.

Sample examination questions:

Questions for the exam " Languages and methods of programming ***"***

1. The concept of a programming language. The syntax and semantics of a language.
2. Ways of implementing languages: compilation, interpretation, mixed approach.
3. Levels of programming languages.
4. Integrated programming systems.
5. Composition of the programming system. Layout and loading of programmes. Program debugging.
6. Classes of programming languages: procedural, object-oriented, functional, logical, scripting languages.
7. Examples of languages.
8. The Delphi programming language. Versions of the language. Basic features. Comparison with other programming languages.
9. Structure of a Delphi programme.
10. Standard data types. Variables, constants, expressions, operations.
11. Type conversion.
12. An assignment operation.
13. Conditional operator.
14. Multiple choice operator.
15. Loop statements in Delphi: with pre-condition, with subsequent condition, with parameter.
16. Nested loops. 17. Operators break, continue, exit.
18. arrays. Handling multidimensional arrays.
19. String as an array of characters.
20. Functions for working with strings. User-created data types: structures, associations, enumerations.
21. Functions in Delphi. The declaration and definition of functions. Function parameters.
22. Methods for passing parameters to a function: by value, by reference, by

pointer.

23. Functions, strings, arrays and structures as function parameters. Use of default arguments. Overloading and function templates.

24. Pointers. Pointers and arrays. Pointers and functions: parameter passing; functions returning pointers; pointers to functions. 25.Storing information in main memory. Allocating memory. Allocating and freeing dynamic memory in Delphi.

26. The operations new and delete. Advantages and disadvantages of dynamic memory management. Typical errors when working with dynamic memory.

27. Dynamic arrays. Creation of one-dimensional and two-dimensional dynamic arrays.

28. Access to the elements of a dynamic array. Dynamic arrays as function parameters.

29. The use of dynamic arrays to solve problems with vectors and matrices that change their size at run time. 30.The concept of a linear list. Bound lists.

31. Types of linked lists: unidirectional, bidirectional, cyclic. A unidirectional list. Adding and removing items to a list.

32. A list implementation in Delphi. A bidirectional list. Adding and removing items to a list. Implementing a list in Delphi. The notions of stack, queue, deck. Implementing them in Delphi. Trees. Using trees for solving tasks.

33. The concept of an algorithm. Properties of an algorithm Time and capacity complexity of an algorithm. Estimating algorithm complexity. Algorithm complexity estimation for basic structures. Classes of algorithms.

34. Sorting algorithms. Internal and external sorting. 35.Direct sorting methods: direct inclusion method, direct selection method, direct exchange method.

36. Fast sorting methods. Fast sorting algorithm. Strategies for selecting a separator. The time complexity of quick sorting.

37. The algorithm for external sorting by simple merging. 38.Search algorithms. Search in linear structures. Binary and interpolation search.

39. The concept of hashing.

40. The concept of recursion. The merits of recursion. Disadvantages of recursive algorithms and how to deal with them. Examples of recursive algorithms. The use of recursion to solve problems.

41. The basic concepts of object-oriented programming (OOP): encapsulation, inheritance, polymorphism.

42. The concepts of class and object.
43. Class fields and methods.
44. Closed and open class elements. Creating objects. 45. Using objects in Delphi programs.
46. Pointers to objects. Passing objects into functions.
47. Objects as return values. Friendly functions. 48. Class constructors and destructors. Copy constructor.
49. Operator overload.
50. Class inheritance. Modes of access to base class elements. 51. Behaviour of constructors and destructors in inheritance.
Multiple inheritance.
52. Virtual methods. Purely virtual methods. Abstract classes

- **10-16** points are awarded to a student if he/she has made several significant errors in the interpretation of basic concepts when answering theoretical questions. Logic and completeness of the answer are markedly flawed. There are notable gaps in the knowledge of basic methods. Theoretical issues are generally presented sufficiently, but with omissions of material. There are fundamental errors in the logic of the answer to a question.

- **1-10 points** are awarded to a student if the answer to the theoretical questions shows a lack of understanding and an extremely incomplete knowledge of the basic concepts and methods. The student could not answer any of the additional questions.

A description of the assessment methodology for the laboratory work:

Evaluation criteria (in points):
For the lab report
- 5 points are awarded to the student if there are no remarks;
- 4 The student is awarded points if there are insignificant remarks;
- 2 A grade is awarded to a student if the results are generally correct, but there are significant comments.

Coursework

The coursework is an assessment competence. The topic is chosen by the student independently and approved by the department.

Suggested topics for coursework

1) Design patterns. Generating patterns-Singleton pattern

2) Design patterns. Generating patterns - Abstract Factory pattern

3) Design patterns. Generating patterns - The Factory Method pattern.

4) Design patterns. Generating patterns - Builder pattern

5) Design patterns. Structural patterns - Adapter pattern.

6) Design patterns. Structural patterns - Facade pattern.

7) Design patterns. Structural patterns - Decorator pattern

8) Design patterns. Structural patterns - Composite pattern.

9) Design patterns. Structural patterns - Substitute (Proxy) pattern.

10) Design patterns. Behavioural patterns - Strategy pattern.

11) Design patterns. Patterns of behaviour - Template Method pattern

12) Design patterns. Behavioural patterns - Mediator pattern

13) Design patterns. Behavioural patterns - Iterator pattern

14) Design patterns. Patterns of behaviour - Observer pattern

15) Design patterns. Behavioural patterns - Visitor pattern

Literature:

1) Teplyakov S. Design patterns on .NET platform. - SPb.: Peter,2015. - 320 p.: ill.

2) Э. Gamma, R. Helm, R. Johnson, J. Vlissides. Methods of object-oriented design. Design patents - SPb: Peter, - P2007.. 366

3) Freeman E. Design patterns. - SPb: Peter, 2003.

5.1. List of basic and additional academic literature required to master the discipline

Basic literature:

1. Konova, E.A. Algorithms and programs. C++ language [Electronic resource] : textbook / E.A. Konova, G.A. Pollak. - Electronic data. - Saint Petersburg : Lan', 2019. - 384 c. - Access mode: https://e.lanbook.com/book/114696.
2. Soldatenko I.S. Practical introduction to C programming language [Electronic resource] : tutorial / I.S. Soldatenko, I.V. Popov. - Electronic data. - Saint Petersburg : Lan', 2018. - 132 c. - Access mode: https://e.lanbook.com/book/109619.

Further reading

3. Zalogova L.A. Fundamentals of object-oriented programming on the basis of C# language [Electronic resource] : tutorial / L.A. Zalogova. - Electronic data. - Saint Petersburg : Lan', 2018. - 192 c. - Access mode: https://e.lanbook.com/book/106731.

5.2. List of resources for the Internet and software required to master the discipline

1. BashSU Electronic Library System https://elib.bashedu.ru/
2. University Library Online digital library system http://www.biblioclub.ru
3. Lanj Library System https://e.lanbook.com
4. Windows Russian8. Windows Professional Russian 8Upgrade. Contract no. of d.10417.06.2013 Licences are perpetual.
5. Microsoft Office Standard Russian2013. Contract no. of d.11412.11.2014 Licences are perpetual.

6. The material and technical resources required for the implementation of the educational process in the discipline

Names of specialized rooms, rooms and laboratories	**Activity form**	**Name of the equipment/software**
1	***2***	***3***
Rooms 501, 502 or any other room according to the current time table	*Lectures*	The board for writing, Projector and screen
Computer classrooms 520a, 521, 522, 525, 523, 426, or any other room according to the current time table	*Laboratory work*	1. Windows 8 Russian. Windows Professional 8 Russian Upgrade. 2. Microsoft Visual Studio Community 2017 3. Python IDLE 3.10.1
Library, reading halls, Computer classrooms 426	*Individual work*	Internet.

Annex No. 1

FEDERAL STATE BUDGETARY EDUCATIONAL INSTITUTION OF HIGHER EDUCATION
"BASHKIR STATE UNIVERSITY"

WORK PROGRAMME CONTENT

Languages and methods of programming for semester 1, 2, 3, 4 full-time

form of training

The work programme is implemented:

Activity	Duration
Total duration of the discipline (CUD / hours)	11/396
Academic hours for the work with instructor	190
lectures	136
seminars	0
laboratory	0
other (consultation in group or individually and other forms of learning activities assuming collaboration of learners with instructor)	3,6
Academic hours for individual work of learners	105,8
Academic hours for preparing to exam/credit test/differentiated credit test (Grading)	96,6

Final grading:

exam in semester 1,2,4

credit in semester 3

No. n/a	Topic and content	Form of study: lectures, workshops, seminars, laboratory work, self-study and workload (in hours)				Basic and supplementary literature recommended to students (list of items)	Assignments for students' independent work	Form of current monitoring (colloquia, quizzes, computer tests, etc.)
		LC	PR/SEM	LR	SR			
1	2	3	4	5	6	7	8	9
	2nd semester	32			31			
1	User-defined data types. Renaming of types (typedef). Enums(enum). Structures (struct,record). Merges (union)	4			4	1-3	Study the following questions; 18-20	Exam
2	Preprocessor directives. The #incluclе directive. The #clefine directive. Conditional compilationdirectives. The #undef directive.	4			4	1-3	Study the following questions; 22-24	Exam
3	Working with files. Writing, reading. Serial access files, text files. files, binary files.	4			4	1-3	Study the following questions; 25-27	Exam
4	Dynamic structures The following are the data: Linear Lists : single and double linked. Linear lists: single- and double-connected lists. The following are the main features of dynamic structures.	4			4	1-3	To study the traces of the The following questions; 28-30	Exam

	wov.							
5	Dynamic structures data. Stacks, queues. Building, adding, removing items.	4			4	1-3	Study the following questions; 31-32	Exam
6	Dynamic structures data. Binary trees. Search tree, perfectly balancedtrees. Tree building, element addition, element removal. Tree traversal.	6			5	1-3	Study the following questions; 33-34	Exam
7	Classes. Class description. Object description. Pointer this. Constructors Copy constructor. Class methods. Static class elements. Static fields. Static methods. Friendly functions and classes. Friendly fun from Friendly class .Destructors	6			6	1-3	Study the following questions; 35-36	Exam
	Total hours:	32			31			
	3-semester	36		18	17.8			
1	Classes. Overloading of operations. Overloading of unary operations. Overloading of binary operations. Overloading of assignment operations. Overloading	4		2	1,8	1-3	lab report no. 1-3	Laboratory work, laboratory work reports, credit.

	The new and delete operations. The following is an example of this:							
	function call operation reloading. Overloading of a function call operation indexing.							
2	Inheritance. Keys of access. Simple inheritance. Virtual methods. Late binding mechanism. Abstract classes. Multiple inheritance. The difference between structures and unions and classes.	4		2	2	1-3	lab report no. 4	Laboratory work, laboratory work reports, credit.
3	Class templates. Creating class templates. Use of class templates. Specialisation of class templates.	4		2	2	1-3	lab report no. 5	Laboratory work, laboratory work reports, credit.
4	Exception handling. A common exception handling mechanism. Syn- taxis of exceptions.	4		2	2	1-3	lab report no. 8	Laboratory work, laboratory work reports, credit.
5	Exception catching. Functional exception catching. Exceptions in constructors and destructors. Hierarchies of exceptions	4		2	2	1-3	lab report no. 8	Laboratory work, laboratory work reports,

6	Streaming classes. Standardthreads. Data formatting. Flags and formatting methods. Manipulators. Methods for exchanging with threads. Errors in streams. File streams. String streams. Streams and types defined by the user	4		2	2	1-3	lab report no. 6	Laboratory work, laboratory work reports, credit.
	the teacher.							
7	Container classes. Container classes Vectors (vectors). De-queues (deque). Lists (list). Stacks. Queue. Priority queue. Associated containers. Dictionaries (tars). Dictionaries with duplicates (multimap). Set Multisets Multisets with duplicates (multiset). Bit sets (multiset). The bitset.	4		2	2	1-3	lab report no. 7	Laboratory work, laboratory work reports, credit.
8	Iterators and functional objects. Iterators . Reverse iterators. Insertion iterators. Threaded iterators. Functional objects. Arithmetic functional objects. Predicates. Negators. Binders. Pointer adapters. functions. Method adapters.	4		2	2	1-3	lab report no. 5	Laboratory work, laboratory work reports, credit.

9	Algorithms. Non-modifying sequence operations. Modifying sequence operations. Algorithms related to sorting.	4		2	2	1-3	lab report no. 7	Laboratory work, laboratory work reports, credit.
	Total hours:	36		18	17,8			0,2
	4-semester	32			20			
1	Creating applications in Microsoft Visual Studio. Creating an MFC application.	4		8	2	1-3	Study the following Questions; 37-38	Examination, coursework report.
2	Working with tests and graphics. Pictures, buttons and cursors in the view window.	4		8	2	1-3	Study the following questions; 39-40	Examination, coursework report.
3	Menu operation. Adding new menu items. Change the operation of menu items. Adding and removing menu items. Adding a text menu.	4		9	2	1-3	Study the following Questions; 41-42	Examination, coursework report.
4	Virtual window, keypad, child window. Scaling the image. Scroll bar handling Handling of keystrokes. Pressing the keys is handled. child window building.	4		9	2	1-3	Study the following Questions; 43-44	Examination, coursework report.

5	The main controlelements of the dialogue windows. Addinga dialogue window. Button, CheckBox Text box (EditControl). The Combo Box (Combo Box). List Box Radio Button. The design elements: Static Text and Group Box	4		12	4	1-3	Study the following questions; 45-47	Examination, coursework report.
	The picture (PictureControl). The slider control (Slider Control) and the counter (Spin Control). Slider Control Counter (Spin Control). The indicator (Progress Control). Hot key (Hot Key). List (List Control). The Tree (Tree Control).							
6	ToolBar and status bar. The tool bar (ToolBar). The status bar (StatusBar). Addingbuttons to the toolbar. Display and hide buttons on the tool bar.	4		9	2	1-3	Study the following Questions; 48-49	Examination, coursework report.

7	Delete and add a button on the toolbar. Add or delete your own toolbar. Addnew fields to the status bar. Change the position and colour of the status bar	4			2	1-3	Study the following questions; 50	Examination, coursework report.
8	The vehicle is equipped with an automatic transmission. Drawing a graphic image. Drawing a graphic using a metafile.	4		9	4	1-3	Study the following questions; 51-52	Examination, coursework report.
	Total hours:	32			20			

Annex No. 2

Rating - discipline plan

Programming languages and methods
field of study [01.03.02] *"Applied mathematics and computer science"*

course semester 1,2

Types of student learning activities	Score for a specific assignment	Number of tasks per semester	points	
			minimum	maximum
Module 1 "Array sorting algorithms"				
Ongoing monitoring				**20**
1. Classroom work			0	10
2. Doing your homework them assignments	5	2	0	10
Module 2 "Functions. Function overload."				
Ongoing monitoring				**25**
1. Classroom work			0	10
2. Doing your homework them assignments	5	3	0	15
Module 3 "Classes."				
Ongoing monitoring				**25**
1. Classroom work			0	10
2. Doing your homework them assignments	5	3	0	15
Attendance (points deducted from the total amount of points gained)				
1. Attendance at a lecture of the lessons			0	-6
2. Visiting a practical (e.g.)			0	-10
Final control				**30**
1. Exam	10	3	0	**30**
TOTAL				**100**

Rating - discipline plan

Programming languages and methods
field of study [01.03.02] *"Applied mathematics and computer science"*

course semester 2,3

Types of student learning activities	Score for a specific assignment	Number of tasks per semester	points	
			minimum	maximum
Module 1 "Inheritance"				
Ongoing monitoring				**20**
1. Classroom work			0	5
2. Laboratory report The work of the	5	3	0	15
Module 2 "Multiple inheritance"				
Ongoing monitoring				**20**
1. Classroom work			0	5
2. Laboratory report The work of the	5	3	0	15
Module 3 "Classroom templates"				
Ongoing monitoring				**20**
1. Classroom work			0	5
2. Laboratory report The work of the	5	3	0	15
Attendance (points deducted from the total amount of points gained)				
1. Attendance at a lecture of the lessons			0	-6
2. Visiting a practical (e.g.)			0	-10
Final control				**40**
1. Credit	10	4	0	**40**
TOTAL				**100**

Rating - discipline plan

Programming languages and methods
Area of training [01.03.02] *"Applied mathematics and computer science"*

course semester 2,4

Types of student learning activities	Score for a specific assignment	Numb er of tasks per semester	points	
			minimum	maximum
Module 1 "Strings. Constructors and String Assignment. Operations"				
Ongoing monitoring				**10**
1. Classroom work			0	5
2. Doing your homework them assignments	5	1	0	5
Module 2 "Lists"				
Ongoing monitoring				**15**
1. Classroom work			0	10
2. Doing your homework them assignments	5	3	0	5
Module 3 "Sets."				
Ongoing monitoring				**15**
1. Classroom work			0	10
2. Doing your homework them assignments	5	3	0	5
Attendance (points deducted from the total amount of points gained)				
1. Attendance at a lecture of the lessons			0	-6
2. Visiting a practical (e.g.)			0	-10
Final control				**60**
1. Exam	10	3	0	**30**
Coursework	30	1		**30**
TOTAL				**100**