SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babes-Bolyai University**
1.2 Faculty	**Faculty of Physics**
1.3 Department	**Department of Physics – Hungarian Line of Study**
1.4 Field of study	**Physics**
1.5 Study cycle	**Master**
1.6 Study programme /Qualification	**Computational Physics / High Energy Physics**

2. Information regarding the discipline

2.1 Name of the discipline	**Object Oriented Programming and Applications in Physics / Programming for HEP**						
2.2 Course coordinator	**Lázár Zsolt Iosif**						
2.3 Seminar coordinator	**Lázár Zsolt Iosif**						
2.4. Year of study	**1**	2.5 Semester	**2**	2.6. Type of evaluation	**E**	2.7 Type of discipline	**Fundamental**

3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	3	Of which: 3.2 course	2	3.3 seminar/laboratory	1
3.4 Total hours in the curriculum	42	Of which: 3.5 course	28	3.6 seminar/laboratory	14
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					30
Additional documentation (in libraries, on electronic platforms, field documentation)					14
Preparation for seminars/labs, homework, papers, portfolios and essays					50
Tutorship					10
Evaluations					4
Other activities:					
3.7 Total individual study hours	108				
3.8 Total hours per semester	150				
3.9 Number of ECTS credits	6				

4. Prerequisites (if necessary)

4.1. curriculum	
4.2. competencies	knowledge of basic algorithms, basic programming competences

5. Conditions (if necessary)

5.1. for the course	board, projector
5.2. for the seminar /lab activities	board, projector, computers

6. Specific competencies acquired

Specific competences	• Using in-depth knowledge of physics, mathematics, and programming in various multi- and inter-disciplinary fields. • Applying computational methods to understand complex scientific phenomena. • Independently apply the achieved knowledge to define and formulate research problems in the computational chemistry and physics fields, use information retrieval, data collection, experiment and/or computer methods to solve such problems. • Advanced skills in modeling molecular systems, statistical and solid-state physics systems using computers. • Ability to critically analyse and evaluate scientific models.
Transversal competences	• Accomplishment of professional tasks in an effective and responsible manner, in compliance with the field-specific legislation and code of ethics. • Ability to work in projects, and also to plan and lead projects. • Effective use of information sources, as well as communication and professional-assisted training resources in both mother tongue and English.

7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	• Students should acquire basic knowledge about modern information technologies and their employment in scientific research
7.2 Specific objective of the discipline	Students will acquire knowledge and understanding of: • fundamental concepts of OOP (object, class, interface, data hiding, polymorphism, inheritance, constructor, etc.) • generalities on OOP methodologies and their specific Java implementations • Unified Modeling Language • designing and implementing graphical user interfaces, events and exceptions • design patterns • basic OOP particularities of C++ • basic OOP particularities of Python • modern tools for scientific computing Students will acquire skills for: • analysis and modeling of complex problems

8. Content

8.1 Course	Teaching methods	Remarks
The software development process, software reuse, programming languages, static and dynamic typing, subtyping, the OpenSource movement. Non-procedural programming. Procedural programming. Object-oriented programming.	oral presentation	
Classes, instances, objects, constructors. Data hiding. Interfaces. Object composition. Inheritance. Polymorphism. Reusability. OOP planning and design.	oral presentation, demonstration	

What is Java? Why Java? Basic syntax. Classes. Constructors. Variables. Operators. Methods. Control flow Primitive data types. Strings. Arrays and vectors. Type casting. Modifiers. Interfaces. Inheritance. Polymorphism Visibility modifiers. Class variables, instance variables. Abstract classes. Other modifiers. Errors and exceptions. Try-Catch-Finally clause. Throwable objects. Catchable vs non-catchable exceptions. Rules and practices.	oral presentation, examples	
What is UML? Use case-, class-, sequence-, activity diagrams.	oral presentation	
What are design patterns? Classification of DP. Examples of DP (Singleton, MVC, etc)	oral presentation, examples	
AWT and Swing. Frames and panels. Controls: buttons, textfields, etc. Layout managers. Event handling: Events and listeners. Windows and mouse listeners. Adapters. Handling the event.	oral presentation, demonstration, examples	
History, scope, advantages, disadvantages. Syntax, pointers, references. Constructors, desctructors, visibility, multiple inheritance, Genericity. Input-Output. Exception and Error handling	oral presentation, example movies	
History, scope, advantages, disadvantages. Syntax, basic features. OOP in Python, Scientific Python, C-Python interface	oral presentation, example movies	
The software development process, software reuse, programming languages, static and dynamic typing, subtyping, the OpenSource movement. Non-procedural programming. Procedural programming. Object-oriented programming.	oral presentation, demonstration	
Classes, instances, objects, constructors. Data hiding. Interfaces. Object composition. Inheritance. Polymorphism. Reusability. OOP planning and design.	oral presentation, method demonstration	
What is Java? Why Java? Basic syntax. Classes. Constructors. Variables. Operators. Methods. Control flow Primitive data types. Strings. Arrays and vectors. Type casting. Modifiers. Interfaces. Inheritance. Polymorphism Visibility modifiers. Class variables, instance variables. Abstract classes. Other modifiers. Errors and exceptions. Try-Catch-Finally clause. Throwable objects. Catchable vs non-catchable exceptions. Rules and practices.	oral presentation, demonstration, examples	
What is UML? Use case-, class-, sequence-, activity diagrams.	oral presentation, demonstration	
What are design patterns? Classification of DP. Examples of DP (Singleton, MVC, etc)	oral presentation, example codes and movies	
AWT and Swing. Frames and panels. Controls: buttons, textfields, etc. Layout managers. Event handling: Events and listeners. Windows and mouse listeners. Adapters. Handling the event.	oral presentation, example movies	
History, scope, advantages, disadvantages. Syntax, pointers, references. Constructors, desctructors, visibility, multiple inheritance, Genericity. Input-Output. Exception and Error handling		
History, scope, advantages, disadvantages. Syntax, basic features. OOP in Python, Scientific Python, C-Python interface		

Bibliography

1. *Java Documentation*, http://java.sun.com/docs/
2. *Unified Modeling Language*, http://www.uml.org
3. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, *Design Patterns sabloane de proiectare*, Teora, 2002
4. MULLER, P., *Introduction to Object Oriented Programming in C++*, http://www.zib.de/Visual/people/mueller/Course/Tutorial/tutorial.html
5. *Python Programming Language*, http://www.python.org
6. Lewis, J. - Loftus, W., *Java Software Solutions: Foundations of Program Design*, Addison-Wesley (2002)

8.2 Laboratory	Teaching methods	Remarks
The tasks will follow the same topic order as on the course	Students will follow step by step the procedure undertaken by the instructor. The screen of the instructor's computer will be mirrored to the projector. Students will present specially selected and agreed upon topics. Students will perfect and present a personal/group software development project wherein they will demonstrate the usage of the learned technologies.	

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

The content of the discipline is consistent with courses of similar content from other foreign academic centers. To adapt to the demands of the labor market, the content of the discipline has been harmonized with the requirements of the pre-university education, research institutes and the business environment.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	End of year examination	Written theoretic and practical exam	40
10.5 Seminar/lab activities	Presentation of a chosen topics	Evaluation of the presentation	15
	Homeworks	Assessing the level of completion and quality of the homework.	15
	Personal/group project	Evaluation of the presentation	30
10.6 Minimum performance standards			
Homework assignments will be turned in every week. Completing and understanding the homework assignments is essential to performing well on the exams and mastering this challenging subject.			

Date

04.05.2023

Signature of course coordinator

[redacted]

Signature of laboratory coordinator

[redacted]

Date of approval

11.05.2023

Signature of the head of department

..[redacted].