

P R E F A C E

These Course Outlines are offered as a general guide to the various courses taught at the Uruguayan American School. It is hoped that the outlines will better enable students, parents, and other interested parties to see at a glance the content and organization of each particular course taught. Additionally the information contained here should enable interested parties in making more viable decisions regarding transfers, both into and out of UAS.

A few Course Outlines are still under development and could not be included here: all levels of French and additional courses in Computer Science specifically. Inquiries regarding these courses should be made to the specific area teacher or to the Coordinator of the Secondary Program.

The Secondary school staff and administration hope you will find these Course Outlines to be of value. Should you have additional concerns or suggestions, we will be pleased to speak with you directly.

COURSE OUTLINE

COURSE TITLE: Information Processing I: Keyboarding Skills, Computer Literacy, Software Evaluations

DURATION: One semester

LEVEL: High School

TEXTBOOKS: Atkinson: "Help with Computer Literacy";
SIU-102: "Computers in Society".

SOFTWARE: Typing Tutor III/Type Attack/Master Type

PRE-REQUISITE: None

COURSE SCOPE: After completion of course student will be able to type with a fluent speed of 30 words per minute, maintaining 80% accuracy. A highly individualized course, Information Processing I offers customized instruction via on-line practice. Performance is monitored weekly and recorded to track student's progress. While developing keyboarding skills, students acquire knowledge regarding the computer's basic applications, as well as an understanding of its limitations. During the initial sessions, students progressively learn the correct fingering using on-line tutorials and customized texts (Typing Tutor). As students demonstrate mastery of finger control technique, computer literacy/composition activities are introduced. Students demonstrating sufficient keyboarding mastery may request a course waiver and proceed to Information Processing II: Modern Office Tools (word processing/data base/worksheet).

COURSE REQUIREMENTS/EXPECTATIONS:

After completion of course student should be able to type with a fluent speed of 30 words per minute, maintaining 80% accuracy. A computerized report will be issued showing current standing. Three composition activities plus three tests on office automation and computer literacy, and four software evaluations will be featured. This is a PASS/FAIL course. Students demonstrating sufficient mastery of keyboarding may request a course waiver and proceed to Information Processing II. Students wishing to do so must however plan to sit for the Computer Literacy Assessment in order to satisfy the required computer literacy units, and will also have to present four software evaluations. Passing factors: timely completion of assignments, average speed as timed by computer program, accuracy as measured by computer program. Passing requirements: produce a score of

30 words-per-minute 80% accuracy on ALL of the following 4-line tests: WORDS test; STANDARD SPEED test; FULL KEYBOARD test. Where applicable, will also be taken into account: spelling, grammar, orderliness.

SYLLABUS

UNIT 1 - The Computer - Today and Yesterday

UNIT 2 - What is a computer?

UNIT 3 - Getting Data into a computer

UNIT 4 - Processing Information

UNIT 5 - Artificial languages

UNIT 6 - What Computers Can -and Cannot- Do

UNIT 7 - Careers with Computers

UNIT 8 - A Look Ahead

SOFTWARE EVALUATIONS: Atkinson's Software Evaluation Forms

COURSE OUTLINE

- COURSE TITLE:** Pascal I (Pascal's for Everybody)
- DURATION:** One semester
- LEVEL:** High School
- TEXTBOOKS:** P. Grogono - "Programming in Pascal"; Carmony, et al - "Problem Solving in Apple Pascal"; Welsh-Elder - "Introduction to Pascal"; Luehrman-Peckham - "Hands on Pascal".
- PRE-REQUISITE:** Information Processing I
- COURSE SCOPE:** A first course on Pascal, "Pascal's for Everybody", introduces, as a second language, the block-structured language developed c. 1971 by Niklaus Wirth. Careful step-by-step explanation of the multiple features and resources of the language is presented, acquainting the student with the new language's specifications. Simple, graded exercises and class-developed applications are included for in-class practice as well as home study. A collected 'type-in' ready-made illustrations series is also incorporated to exemplify statements' usage. Process logic and analysis are also introduced featuring modelling situations. This offering is ALSO offered as a FIRST artificial language course.

COURSE REQUIREMENTS/EXPECTATIONS:

A minimum of twenty (20) satisfactorily completed projects or 70 project points has to be achieved throughout the course. A catalog containing the projects' titles and point values is posted at the beginning of the term for student reference. There will be four (4) (30% of the grade) tests on programming and computer literacy. All accepted projects MUST be handed in using 'official' paper, indicating name, grade, date and project id. Programming projects account for 70% of the grade. Where considered convenient, a written/oral report may be requested. Individual work and 'team'-projects are NOT synonyms. Late entries may not be accepted. Self-prompted projects are greatly encouraged and will add to the final grade. However, be sure to check out project idea with your instructor and get clearance before going ahead. ALL programs will be tested by running the program "on the spot". ALL projects should be in your notebook, which MUST be kept in TOP CONDITION and available for inspection at all times. The notebook/folder must contain an index of ALL projects, handouts and activities, as

well as a list of completed projects. Program print-outs can be pasted onto the notebook to avoid manual recopying. Although a Red Dot System Chart is posted and updated accordingly to reflect/help student performance, the final evaluations are made upon the accepted projects records fed to the course's data base. It is important to CAREFULLY SAVE/KEEP ALL YOUR COMPLETED WORK AS EVIDENCE in case of computer breakdown.

EXTENSION:

The following items are also included in the course: decision tables; process logic analysis & presentation; diagrams and flowcharting guidelines; elementary logic; study of algorithms; step-wise refinement/top-down design.

SYLLABUS

- UNIT 1 - I/O statements: WRITE, Writeln**
special features: chr(12) to erase the screen
sending output to the printer
application projects (ids): 604
- UNIT 2 - Primitive data structures; simple variables & constants**
I/O statements: READ, READLN
arithmetic operators + * - /
examples involving all four operations
special features: system functions: square root
exp. function
trig. lines/abs
application projects (ds): 600, 602, 603, 605
- UNIT 3 - The IF statement construct;**
application projects (ids): 607, 612, 613
- UNIT 4 - The WHILE statement loop construct;**
application projects (ids): 607, 612, 613
- UNIT 5 - THE FOR statement loop construct; applications**
application projects (ds): 607, 612, 613
- UNIT 6 - The REPEAT/UNTIL statement loop construct; applications**
application projects (ids): 609-611
- UNIT 7 - The TYPE declaration**
application projects (ids): 602
- UNIT 8 - data structures: numeric arrays**
application projects (ids): 615, 629
- UNIT 9 - data structures: string arrays**
application projects (ids): 614, 616

Students also learn how to edit, compile, run and file Pascal programs using the facilities (ie. editor, compiler, filer) provided by a suitable operating system. Currently UCSD Apple Pascal System and Turbo Pascal are being used. Further information can be found in the latest edition of the Computer Department handbook.

COURSE OUTLINE

- COURSE TITLE:** Pascal II/AP
Pascal III/AP
- DURATION:** One semester each
- LEVEL:** High School
- TEXTBOOKS:** P. Grogono - "Programming in Pascal";
Balestie - AP Class Handbook
- REFERENCE:** Welsh-Elder - "Introduction to Pascal";
Wirth - "Algorithms + Data = Programs";
Schneider-Bruell - "Advanced Programming
and Problem Solving with Pascal"; Donald
Knuth - "The Art of Computer Programming"
Vol. 1-3; Sedgewick "Algorithms"; Tenenbaum
& Augenstein - "Data Structures Using
Pascal"; Standish - "Data Structure
Techniques"; Aho, Hopcroft & Ullman's "Data
Structures & Algorithms"; Gane & Sarson -
"Structured Systems Analysis".
- PRE-REQUISITE:** Information Processing I / Pascal I
- COURSE SCOPE:** Intended for students already familiar with
the language's syntax, Pascal II & III
cover further developments and more
'serious' applications. A broader scope of
the student's responsibility share in a
software project team is called for, as
opposed to individual 'line' coding; that
is, the process of writing statements in a
particular computer language. A more
global approach is aimed at, exploring the
entire steps and stages involved in solving
a problem using computer equipment: problem
definition ('problem statement'); solution
methods selection ('analysis'); program
testing and verification ('running and
testing'); system maintenance and
documentation techniques. Pascal III
provides ALL the necessary programming
background information/training when
preparing for the Advanced Placement test
in Computer Science; stacks, queues, trees
(binary, balanced, threaded, 8, heaps),
tries, hashing tables, as well as garbage
collectors, searching, sorting, traversals,
and external file organization.
- COURSE REQUIREMENTS/EXPECTATIONS:**

A minimum of twenty (20) satisfactorily
completed projects or 120 (Pascal II) / 200
(Pascal III) project points has to be
achieved throughout the course. A catalog
containing the projects' titles and point
values is posted at the beginning of the
term for student reference. There will be
four (4) (30% of the grade) tests on
programming and computer literacy. All
accepted projects MUST be handed in using

'official' paper, indicating name, grade, date and project id. Programming projects account for 70% of the grade. Where considered convenient, a written/oral report may be requested. Individual work and 'team'-projects are NOT synonyms. Late entries may not be accepted. Self-prompted projects are greatly encouraged and will add to the final grade. However, be sure to check out project idea with your instructor and get clearance before going ahead. ALL programs will be tested by running the program "on the spot". ALL projects should be in your notebook, which MUST be kept in TOP CONDITION and available for inspection at all times. The notebook/folder must contain an index of ALL projects, handouts and activities, as well as a list of completed projects. Program print-outs can be pasted onto the notebook to avoid manual recopying. Although a Red Dot System Chart is posted and updated accordingly to reflect/help student performance, the final evaluations are made upon the accepted projects records fed to the course's data base. It is important to CAREFULLY SAVE/KEEP ALL YOUR COMPLETED WORK AS EVIDENCE in case of computer breakdown.

EXTENSION:

The following items are also included in the course: decision tables; process logic analysis & presentation; diagrams and flowcharting guidelines; elementary logic; study of algorithms; step-wise refinement/top-down design; numerical analysis; automata theory; coding theory; operations research; systems architecture.

SYLLABUS

- UNIT 1 - General background about computers; the concept of an algorithm; editing and executing programs; specific machine dependent information
- UNIT 2 - Warm up activities writing simple programs; numeric data, variables, expressions, assignments simple I/O
- UNIT 3 - Conditionals; relational and logical operators; IF statements
- UNIT 4 - Loops; FOR and WHILE loop constructs
- UNIT 5 - Modular programming; procedures and functions
- UNIT 6 - Parameter passing, recursion, call by value, call by reference
- UNIT 7 - Arrays; vectors, matrices; applications
- UNIT 8 - Searching; linear and binary search
- UNIT 9 - Sorting; selection, exchange, and insertion sort
- UNIT 10 - Efficient sorting techniques: merge/sort; quicksort

UNIT 11 - Data Structures: review primitive types, arrays, introduce sets and records

UNIT 12 - String data and operations

UNIT 13 - Text files and applications

UNIT 14 - Review of programming methodology; program style; top-down design

UNIT 15 - Program correctness; testing; verification

PASCAL III

UNIT 16 - Numerical methods; numerical errors; roots of a function eg. roots of a cubical polynomial

UNIT 17 - Simulation; deterministic simulation; random processes; discrete event simulation; Monte Carlo methods.

UNIT 18 - Linked lists; pointers; list manipulation

UNIT 19 - Queues; simulation of waiting lines; statistics

UNIT 20 - Binary search trees; traversal; tree sort and search heaps; heapsort.

UNIT 21 - Stacks; stack and recursion

UNIT 22 - Hashing; overflow handling; hashing functions

UNIT 23 - Symbolic manipulation of expressions; syntax trees; bottom-up/top-down parsing; compiler construction

UNIT 24 - Game theory; linear programming; game trees