Southend High School for Girls

Assessment Policy for Computer Science

Our assessment policy and procedures are underpinned by three key questions :

1. Where is the learning going?
2. Where is the learner now?
3. How does the learner get there?
   (Dylan Wiliam)

Our students should be able to answer these two questions:

1. What am I doing well in the subject?
2. What do I need to do to improve my work in the subject?

At SHSG we see assessment, in all its forms, as an integral part of teaching and learning and as such it is inextricably linked to our curriculum.

We use three broad overarching forms of assessment at Southend High School for Girls:

1. Diagnostic assessment – assessment used to determine what students already know (usually at the start of a lesson / unit)
2. Formative assessment (responsive teaching) – assessment used during the learning process to provide feedback and encourage students to act upon it to make improvements
3. Summative assessment (in-school summative assessment and nationally standardised summative assessment) – used at the end of the learning process as a measure of where students are in their learning.

KS3

By the end of KS3 students should be able to demonstrate a clear understanding of the fundamental concepts and principles of computer science, including computational thinking, algorithms, data representation, and computer systems.

The student should be able to apply their knowledge of computer science to solve problems and complete tasks using appropriate software and programming languages. They should be able to design, write, and debug programs, as well as understand and modify existing code.

The student should demonstrate creativity by generating solutions, designing aesthetically pleasing user interfaces or graphics, and thinking outside the box when confronted with problems or challenges in computer science.

KS4

By the end of KS4 students should be able to be able to understand and use computational thinking principles such as decomposition, pattern recognition, abstraction, and algorithmic design to solve problems.

Students should be proficient in at least one programming language, such as Python. They should be able to write code to solve problems, understand and use different data types, variables, operators, control structures (e.g., loops and conditionals), functions/subroutines, and arrays/lists.

Students should have a good understanding of how computers work at a hardware level. This includes knowledge of binary representation, logic gates, Boolean algebra, computer memory (RAM and ROM), CPU architecture (fetch-decode-execute cycle), secondary storage devices (e.g., hard drives and solid-state drives), and input/output devices.

Students should have a good understanding of software including system software, application software, utilities. They should understand the functions and characteristics of operating systems.

Students should understand how data is represented in computers using binary code. This includes knowledge of binary numbers, hexadecimal numbers, ASCII/Unicode character encoding, images (pixels), sound (sample rates), and compression techniques.

Students should understand what algorithms are. They should be able to write, and correct algorithms for different problem-solving scenarios using pseudocode or flowcharts. They should know how to use the standard search and sorting algorithms.

Students should understand computer networks including LANs (Local Area Networks) and WANs (Wide Area Networks). They should know about network topologies (e.g., star or mesh), network protocols (e.g., TCP/IP), IP addressing), DNS (Domain Name System), firewalls, encryption methods, and the concept of client-server architecture.

Students should be aware of common cybersecurity threats and measures to protect computer systems and networks. This includes knowledge of malware (e.g., viruses, worms, or Trojans), social engineering (e.g., phishing or scams), encryption techniques, secure coding practices, password security, and ethical considerations related to cybersecurity.

Students should understand the ethical, legal, and environmental implications of using computer systems. This includes knowledge of copyright laws, data protection regulations

KS5

By the end of KS5 students should be able to demonstrate knowledge, understanding and ability to application in the following areas:

Component 01: Computer systems

- The characteristics of contemporary processors, input, output and storage devices
- Types of software and the different methodologies used to develop software
- Data exchange between different systems
- Data types, data structures and algorithms
- Legal, moral, cultural and ethical issues.

Component 02: Algorithms and programming

- What is meant by computational thinking (thinking abstractly, thinking ahead, thinking procedurally etc.)
- Problem solving and programming – how computers and programs can be used to solve problems
- Algorithms and how they can be used to describe and solve problems.

Component 03: Programming project

Students are expected to apply the principles of computational thinking to a practical coding programming project. They will analyse, design, develop, test, evaluate and document a program written in a suitable programming language. The project is designed to be independently chosen by the student and provides them with the flexibility to investigate projects within the diverse field of computer science. We support a wide and diverse range of languages.

Diagnostic assessment

- Computer based topic assessment quizzes
- Cold calling with whiteboards (KS3/4)
- Review of individual progress programming assignments

Formative assessment methods

- Computer based topic assessment quizzes
- Programming assignment progress scores

Summative assessment

- Computer based topic and cumulative assessment quizzes
- Programming assignment progress scores
- Year 9 website project (KS3)

- Past paper questions (KS4 /KS5)
- Coursework project (KS5)

Marking and Feedback

Marking and Feedback Codes

When providing written feedback in exercise books to ensure consistency across the school particularly for literacy the following codes should be used above the relevant word /section:

✓	good point
X	incorrect or wrong point
SP	spelling error which needs correction
P	punctuation error which needs correction
GR	grammatical error which needs correction
/	start a new sentence
//	start a new paragraph
??	the point is not clear
WW	wrong word
^	missing word
[ ]	this part needs rewording
+1	academic achievement point

<u>Coding feedback can be done using the following abbreviations:</u>

These should be written in the Teams assignment feedback box, or directly to the programming document if using Replit.com.

XW - excellent work

GW - good work overall

ME - more effort needed / more work expected

TU - attempt more of the tutorials

C - more comments needed to explain key aspects of the program

NF - not finished, please complete

CL - check program logic

SE - syntax error

SP - spelling errors

VR – variable errors

IN – indentation error

LO – loop error

OP - output not shown

TE - have you tested the program?

RU - try the code for another purpose

EW - try the extension work next time

EX - explain the code/topic more

EM- Show error messages in your work

RC - research the code further and try other python coding related to this

OH - come along to open house to get help

AP - 1 achievement point awarded

Presentation

- All work should have a date on the right hand side, written in full and underlined with a ruler (e.g. 12th September 2023.
- All work will have a title / heading which is underlined with a ruler
- All work should have CW/HW written in the top left-hand margin
- Only black or blue ink should be used for writing with the exception of student responses to feedback (as indicated by individual department policies)
- All diagrams / graphs should be done in pencil.
- All work should be set out neatly.
- The majority of written CS work is recorded into Onenote classnotebook (KS3) in the student's own class notes section.
- For KS4 and KS5 students make notes in a cornell notes style.
- KS4 and KS5 students will use word, onenote and powerpoint to document their progress on class and homework tasks.

Recording and Monitoring of Assessment

- KS3 data is stored by the individual teacher either in excel or Teams assignments.
- KS3 quiz data is stored and accessible across the department in Educake
- KS3 programming project progress is stored in Amazon turinglab.

- There is a shared departmental tracking Excel sheet.
- KS4 quiz data is stored and accessible across the department in Educake
- KS4 practice examination question data is stored and accessible across the department in Smart revise
- KS5 practice examination question data is stored and accessible across the department in Smart revise
- KS4 programming progress is stored in replit.com teams
- KS5 programming progress is stored in replit.com teams
- Regular download and analysis of the Educake, Turinglab, Smart Revise and replit data is done by teachers and HOD.