H2 Computing (9569)

WA Revision Notes (Condensed)

Social, Ethical, Legal & Economic Issues

SELE framework: Structure discuss answers around Social, Ethical, Legal, and Economic angles.

Social: Digital divide, social isolation, info overload, job displacement, changing norms.

Ethical: Privacy vs convenience, algorithmic bias, autonomous liability, AI-generated IP, whistleblowing.

Legal: MCA (unauthorised access), POFMA (falsehoods), POHA (harassment/doxxing), PDPA (data misuse), piracy, hacking.

Economic: Job displacement, new tech jobs, Big Tech monopolies (GAFA), gig economy, cybersecurity costs.

Singapore Acts & Professional Ethics

Act Quick-ID

Act	Triggers
MCA	Hacking, unauthorised access, password theft, data modification
POFMA	False statement of fact online (public interest) – NOT opinion
POHA	Harassment, cyberbullying, stalking, doxxing, threats
PDPA	Collection/use/disclosure of personal data without consent

! Multiple acts apply

One scenario can trigger 2+ acts. Hack hospital = MCA. Post stolen patient records online = POHA + PDPA. Identify ALL applicable acts.

2.1 Computer Misuse Act (MCA)

Sec 3 (unauth access, max \$5k+2yr), Sec 4 (access+intent to offend, max \$50k+10yr), Sec 5 (unauth modification, max \$10k+3yr), Sec 6 (unauth use/interception), Sec 7 (obstructing access), Sec 8 (supplying hacking tools). Covers direct and indirect access.

2.2 POFMA

False statements of fact (not opinion/satire). Correction direction (notice alongside, no removal), Stop communication direction, Disabling direction (ISP block). Max: \$50k+5yr (indiv), \$500k (corp).

2.3 POHA

Sec 3 (harassment/alarm/distress), Sec 5 (fear of violence), Sec 7 (unlawful stalking). Also doxxing. Civil remedies: Protection Orders (PO), Enhanced PO. Extends to overseas offenders if victim in Singapore.

2.4 PDPA

9 obligations: Consent, Purpose limitation, Notification, Access & correction, Accuracy, Protection, Retention limitation, Transfer limitation, Breach notification. DNC Registry. Fines: up to 10% turnover or \$1M.

2.5 Professional Ethics

Competence, Integrity, Confidentiality, Accountability, Public interest, Whistleblowing.

Database Management Systems

3.1 Key Terms

PK: unique, NOT NULL, one per table. **Composite key**: 2+ fields as PK. **FK**: references PK of another table (referential integrity). **Secondary key**: alternative search key.

3.2 Redundancy & Normalisation

Data redundancy = wasted storage + update/insert/delete anomalies. Data dependency = partial or transitive. Fixed via normalisation.

1NF: atomic cells, no repeating groups, PK exists.
2NF: 1NF + no partial dependencies (non-key depends on entire PK).
3NF: 2NF + no transitive dependencies (non-key depends directly on PK).

Example

Normalisation Example Enrolment(SID, SName, CCode, CTitle, Lecturer, Grade)
PK = {SID, CCode}. Partial: SName depends on SID only; CTitle, Lecturer depend on CCode only.
Split into: Students(SID, SName), Courses(CCode, CTitle, Lecturer), Enrolment(SID, CCode, Grade).
Check for transitive deps (e.g., Lecturer → Dept → DeptHead) to reach 3NF.

3.3 SQL vs NoSQL

Dimension	SQL	NoSQL
Schema	Rigid, predefined	Flexible, dynamic
Scaling	Vertical (scale-up)	Horizontal (scale-out)
Consistency	Strong (ACID)	Eventual (BASE)
Query	Standardised SQL	API/vendor-specific
Model	Tables/relations	Documents/graphs/KV
Examples	MySQL, SQLite	MongoDB, Redis

Python Programming (Ch 1–6)

4.1 Data Types

int, float, bool, str (immutable), list (mutable ordered), dict (key-value), tuple (immutable).
Critical: `input()` returns str – cast with `int()/float()` for arithmetic.

4.2 Operators

Arithmetic: `+ - * / % ** //`. Relational: `== != > < >= <=`. Logical: `and or not` (short-circuit). Membership: `in not in`.

4.3 Selection

```
if score >= 80:
    grade = "A"
elif score >= 70:
    grade = "B"
else:
    grade = "F"
```

4.4 Iteration

```
while count < 5:
    print(count)
    count += 1

while True:              # sentinel
    val = input("q to quit: ")
    if val == "q": break

for i in range(5):       # 0..4
    print(i)
for i in range(2,8):     # 2..7
    print(i)

for item in lst:
    print(item)
for i, v in enumerate(lst):
    print(i, v)
```

4.5 Lists

Create: `lst = []`, `lst = [1,2,3]`. Read: `lst[2]`, `lst[1:4]`. Update: `lst[2]=10`. Append: `lst.append(4)`. Insert: `lst.insert(idx,val)`. Remove: `lst.remove(val)` or `lst.pop(idx)`. Clear: `lst.clear()`. Sort: `lst.sort()`. Membership: `if val in lst:`.
Aliasing: `b = a` does NOT copy. Use `b = a.copy()` or `b = a[:]`.

4.6 Dictionaries

Create: `d = {}`, `d = {key:val}`. Read: `d[key]` or `d.get(key, default)`. Update/Insert: `d[key]=val`. Delete: `del d[key]` or `d.pop(key)`. Check: `if key in d:`. Keys/Values/Items: `d.keys()`, `d.values()`, `d.items()`.

4.7 Strings

Immutable – `s[0] = "x"` raises TypeError.
Methods: `.upper()`, `.lower()`, `.isdigit()`, `.isalpha()`, `.strip()`, `.split(d)`, `d.join(lst)`, `.startswith(p)`, `.find(sub)`, `.replace(o,n)`.

4.8 Functions

Local scope by default. Use `global x` to modify global variable.
Trap: `def f(lst=[]):  lst.append(1)` accumulates across calls. Use `None` as default.

4.9 File I/O

```
with open("data.txt", "r") as f:
    content = f.read()        # entire file
    for line in f:            # line by line
        line = line.strip()
with open("out.txt", "w") as f:
    f.write("Hello\n")
```

Modes: `"r"` (read), `"w"` (write/overwrite), `"a"` (append), `"r+"` (read+write).

4.10 Exception Handling

```
try:
    num = int(input("Number: "))
    result = 10 / num
except ValueError:
    print("Not an integer!")
except ZeroDivisionError:
    print("Cannot divide by zero!")
else:
    print("No errors!")    # runs if no exception
finally:
    print("Always runs")   # cleanup
```

Common exceptions: ValueError, TypeError, ZeroDivisionError, IndexError, KeyError, FileNotFoundError.

4.11 Validation & Testing

Validation (computer checks): presence, type, range, length, format, check digit, existence.
Verification (human checks vs source document).
Error types: Syntax (caught before run), Runtime (exceptions), Logic (wrong output – hardest).
Test cases: Normal (valid), Boundary (edges), Erroneous (invalid).

Pseudocode

Note: No specific syntax required – marks for logical correctness.

5.1 Selection

```
IF score >= 80 THEN
    grade <- "A"
ELSE IF score >= 70 THEN
    grade <- "B"
ELSE
    grade <- "F"
END IF
```

5.2 Iteration

```
FOR i <- 1 TO 10
    OUTPUT i
END FOR

count <- 0
WHILE count < 5
    OUTPUT count
    count <- count + 1
END WHILE
```

5.3 Arrays

```
numbers <- [10, 20, 30]
x <- numbers[2]           # 0-indexed

FOR i <- 0 TO LENGTH(arr)-1
```

```
    OUTPUT arr[i]
END FOR
```

5.4 Functions

```
FUNCTION factorial(n)
    IF n <= 1 THEN
        RETURN 1
    ELSE
        RETURN n * factorial(n-1)
    END IF
END FUNCTION

PROCEDURE greet(name)
    OUTPUT "Hello, " + name
END PROCEDURE
```

5.5 Trace Tables

Columns = all variables + conditions. New row per change.

Example

Trace Table Example Algorithm: x=1, total=0; WHILE x <= 4: total+=x; x+=2; OUTPUT total.

Iter	x	total	Output
Init	1	0	
1	3	1	
2	5	4	
Exit	5	4	4

Exam Techniques

6.1 Command Words

State/Identify (1m, brief answer), Describe (features), Explain (how/why), Discuss (both sides + judgement), Compare (similarities + differences), Contrast (differences only), Evaluate (judgement with evidence), Distinguish (what differs), Illustrate (examples/diagrams).

6.2 Timing (Paper 1: 3h, 100 marks)

~1.8 min per mark. 12-mark Q ≈ 22 min. 15-mark Q ≈ 27 min. Leave 10 min for review.

6.3 Top 10 Exam Errors

1. Missing applicable acts – cross-check MCA/POFMA/POHA/PDPA
2. Confusing validation vs verification
3. Normalisation without stating WHICH dependency eliminated
4. Forgetting to cast `input()` – returns str
5. Mutable default argument trap – use None, not []
6. List aliasing – `b = a` does not copy
7. Strings immutable – `s[0] = "x"` raises TypeError
8. Fixating on pseudocode syntax – marks for logic
9. Trace tables missing columns/variables
10. Forgetting boundary test cases

Good luck for your WA!