

NUST NET practice: computer science (30 MCQs)

This practice set has 30 original multiple-choice questions for students taking the NUST Entry Test (NET) in the computer science group. It is written at FSc/ICS (HSSC) computer science level. It covers number systems and conversions, binary arithmetic, logic gates and Boolean algebra, computer architecture (CPU, memory and buses), operating systems, networking basics, databases, and programming fundamentals in C. NUST publishes a topic list for the NET, not numbered syllabus sections, so topics are named here instead of section numbers. Check the current NUST admissions guide for the latest topic list. An answer key and worked solutions come after the questions.

How to use this paper

- **Marking:** give yourself 1 mark for each correct answer and 0 for a wrong or blank answer. Always check the current NUST admission guide for the official marking scheme, including whether wrong answers lose marks.
- **Timing:** allow 30 minutes, which is about one minute per question. The code-tracing questions (21 to 30) and the scheduling question (14) take longest, so do the quick recall questions first and come back to them.
- **Code questions:** assume every C snippet sits inside `int main()` with `#include <stdio.h>` at the top of the file and compiles without errors. The exception is Q28, which is given as a complete program. Line breaks are shown with separate lines; the indentation is not important.
- **Boolean notation:** A' means NOT A, $A \cdot B$ (or AB) means A AND B, and $A + B$ means A OR B.
- **Method:** trace code on paper by writing each variable's value after every line. Guessing the output "by eye" is the most common way to lose marks in this section.

Section 1: Number systems and binary arithmetic

1. What is $(10110110)_2$ in decimal? [1]

- A. 178
- B. 182
- C. 186
- D. 190

2. Convert $(3F7)_{16}$ to octal. [1]

- A. $(1767)_8$
- B. $(1777)_8$
- C. $(1667)_8$
- D. $(7671)_8$

3. What is the 8-bit two's complement representation of -45 ? [1]
- A. 11010010
 - B. 10101101
 - C. 11010100
 - D. 11010011
4. The unsigned 8-bit numbers 10110110 and 01101101 are added in an 8-bit register. What is the result? [1]
- A. 00100011, with a carry out of 1
 - B. 11100011, with no carry out
 - C. 00100011, with no carry out
 - D. 00100001, with a carry out of 1
5. What range of integers can be stored in 8 bits using two's complement? [1]
- A. 0 to 255
 - B. -127 to $+127$
 - C. -128 to $+127$
 - D. -127 to $+128$

Section 2: Logic gates and Boolean algebra

6. Which two-input gate gives an output of 0 only when both inputs are 1? [1]
- A. AND
 - B. NOR
 - C. XOR
 - D. NAND
7. Simplify the Boolean expression $(A + B)(A + C)$. [1]
- A. $AB + C$
 - B. $A + BC$
 - C. $A + B + C$
 - D. ABC
8. Which expression equals the complement of $(A \cdot B + C)$? [1]
- A. $A'B' + C'$
 - B. $(A' + B') \cdot C'$
 - C. $(A + B) \cdot C'$
 - D. $A' \cdot B' \cdot C'$
9. A circuit has output $Y = A \cdot B' + A' \cdot C$. Which input combination makes $Y = 1$? [1]
- A. $A = 0, B = 0, C = 0$

- B. $A = 1, B = 1, C = 0$
- C. $A = 1, B = 1, C = 1$
- D. $A = 0, B = 1, C = 1$

Section 3: Computer architecture

10. Which CPU register holds the address of the next instruction to be fetched? [1]

- A. Memory Address Register (MAR)
- B. Program Counter (PC)
- C. Instruction Register (IR)
- D. Accumulator (AC)

11. A processor has a 20-bit address bus and byte-addressable memory. What is the largest amount of memory it can address directly? [1]

- A. 1 MB
- B. 20 KB
- C. 20 MB
- D. 4 GB

12. Which statement about the system bus is correct? [1]

- A. The data bus is bidirectional, and the address bus carries addresses from the CPU to memory and I/O.
- B. The width of the address bus sets the word size of the CPU.
- C. The control bus carries the data values being read from memory.
- D. Widening the data bus increases the number of memory locations that can be addressed.

13. Which type of memory loses its contents when the power is switched off? [1]

- A. ROM
- B. EEPROM
- C. Flash memory
- D. DRAM

Section 4: Operating systems

14. Three processes arrive at time 0 in the order P1, P2, P3. They need 10 ms, 3 ms and 6 ms of CPU time. The operating system uses round-robin scheduling with a time quantum of 4 ms, and context-switch time is ignored. At what time does P3 finish? [1]

- A. 11 ms
- B. 15 ms
- C. 17 ms
- D. 19 ms

15. Which of these is **not** a function of an operating system? [1]

- A. Allocating main memory to running programs
- B. Scheduling processes on the CPU
- C. Managing files and directories on storage devices
- D. Translating C source code into object code

Section 5: Networking

16. Which device works at the network layer and forwards packets between different networks using IP addresses? [1]

- A. Hub
- B. Switch
- C. Router
- D. Repeater

17. How long is an IPv4 address? [1]

- A. 32 bits
- B. 48 bits
- C. 64 bits
- D. 128 bits

18. A 5 MB file is sent over a link with a data rate of 10 Mbps. Take $1 \text{ MB} = 10^6$ bytes and $1 \text{ Mbps} = 10^6$ bits per second, and ignore protocol overheads. What is the minimum transfer time? [1]

- A. 0.5 s
- B. 4 s
- C. 40 s
- D. 50 s

Section 6: Databases

19. Which statement best describes a foreign key? [1]

- A. A field in one table whose values refer to the primary key of another table
- B. A field that must be different in every record of its own table and can never be empty
- C. A combination of every field in a table used to sort records
- D. A password that protects a table from unauthorised access

20. The table **Students** contains these records:

Name	Marks	City
Ali	82	Lahore
Sana	68	Lahore

Bilal	75	Karachi
Hira	91	Lahore
Usman	70	Lahore

How many rows does this query return? [1]

```
SELECT Name FROM Students WHERE Marks > 70 AND City = 'Lahore';
```

- A. 1
- B. 2
- C. 3
- D. 4

Section 7: Programming fundamentals in C

21. What is the output of this code? [1]

```
int a = 17, b = 5;
printf("%d %d", a / b, a % b);
```

- A. 3 2
- B. 3.4 2
- C. 3 3
- D. 2 3

22. What is the output of this code? [1]

```
int x = 5;
int y = x++ + 2;
printf("%d %d", x, y);
```

- A. 5 7
- B. 6 8
- C. 6 7
- D. 5 8

23. What value is stored in r? [1]

```
int r = 2 + 3 * 4 % 7;
```

- A. 0
- B. 14
- C. 7
- D. 6

24. What is the output of this code? [1]

```
int i, count = 0;
for (i = 1; i <= 20; i += 3)
    count++;
printf("%d", count);
```

- A. 6
- B. 7
- C. 20
- D. 8

25. What is the output of this code? [1]

```
int n = 1234, s = 0;
while (n > 0) {
    s += n % 10;
    n /= 10;
}
printf("%d", s);
```

- A. 4321
- B. 1234
- C. 10
- D. 24

26. What is the output of this code? [1]

```
int k = 2;
switch (k) {
    case 1: printf("A");
    case 2: printf("B");
    case 3: printf("C"); break;
    default: printf("D");
}
```

- A. BCD
- B. B
- C. ABC
- D. BC

27. What is the output of this code? [1]

```
int a[5] = {4, 9, 2, 7, 5};
int i, s = 0;
for (i = 0; i < 5; i += 2)
```

```
s += a[i];  
printf("%d", s);
```

- A. 11
- B. 16
- C. 27
- D. 18

28. What is the output of this program? [1]

```
#include <stdio.h>  
void change(int x) {  
    x = x * 10;  
}  
int main() {  
    int n = 4;  
    change(n);  
    printf("%d", n);  
    return 0;  
}
```

- A. 40
- B. 0
- C. 4
- D. An unpredictable (garbage) value

29. How many asterisks does this code print? [1]

```
int i, j;  
for (i = 1; i <= 4; i++)  
    for (j = 1; j <= i; j++)  
        printf("*");
```

- A. 4
- B. 8
- C. 16
- D. 10

30. What is the output of this code? [1]

```
float f = 7 / 2;  
printf("%.1f", f);
```

- A. 3.5
- B. 3.0

- C. 4.0
- D. 3

Answer key

Question	Answer	Question	Answer
1	B	16	C
2	A	17	A
3	D	18	B
4	A	19	A
5	C	20	B
6	D	21	A
7	B	22	C
8	B	23	C
9	D	24	B
10	B	25	C
11	A	26	D
12	A	27	A
13	D	28	C
14	C	29	D
15	D	30	B

Answer spread: A = 8, B = 8, C = 7, D = 7.

Worked solutions

1. Answer B (182)

Write the place value above each bit, from 2^7 down to 2^0 :

27	26	25	24	23
1	0	1	1	0

27	22	21	20
1	1	1	0

Add the place values that have a 1: $128 + 32 + 16 + 4 + 2 = 182$.

2. Answer A ((1767)₈)

Go through binary. Each hex digit becomes 4 bits: 3 = 0011, F = 1111, 7 = 0111, so $(3F7)_{16} = 0011\ 1111\ 0111$.

Starting from the right, regroup into 3-bit groups: 001 111 110 111.

Convert each group to one octal digit: 001 = 1, 111 = 7, 110 = 6, 111 = 7. The result is $(1767)_8$.

Check in decimal: $3 \times 256 + 15 \times 16 + 7 = 1015$, and $1 \times 512 + 7 \times 64 + 6 \times 8 + 7 = 512 + 448 + 48 + 7 = 1015$. ✓

3. Answer D (11010011)

Step 1: write +45 in 8 bits. $45 = 32 + 8 + 4 + 1$, which gives 001011101.

Step 2: invert every bit (one's complement) to get 11010010.

Step 3: add 1 to get 11010011.

Check: the MSB has weight -128, so $-128 + 64 + 16 + 2 + 1 = -45$. ✓ Option A is the one's complement only; students who forget to add 1 choose it.

4. Answer A (00100011, carry out 1)

Convert both numbers: $10110110 = 182$ and $01101101 = 109$. The sum is $182 + 109 = 291$.

The largest value an 8-bit register can hold is 255, so the sum overflows the register. $291 - 256 = 35$, and the 256 comes out as a carry of 1 from the most significant bit.

$35 = 32 + 2 + 1 = 00100011$. The register holds 00100011 and the carry out is 1.

You get the same result by adding column by column from the right and carrying 1 whenever a column adds up to 2 or 3.

5. Answer C (-128 to +127)

With n bits, two's complement covers -2^{n-1} to $2^{n-1} - 1$. For $n = 8$, that is $-2^7 = -128$ up to $2^7 - 1 = +127$. The range is not symmetric because zero has only one pattern (00000000), which leaves one extra negative value ($10000000 = -128$). Option A is the unsigned range. Option B is the range for sign-and-magnitude or one's complement.

6. Answer D (NAND)

NAND is NOT(AND). AND gives 1 only when both inputs are 1, so NAND gives 0 only in that case and 1 in the other three rows. NOR gives 1 only when both inputs are 0. XOR gives 0 whenever the two inputs are equal, so it gives 0 for both 00 and 11.

7. Answer B ($A + BC$)

Expand: $(A + B)(A + C) = A \cdot A + A \cdot C + B \cdot A + B \cdot C$.

$A \cdot A = A$ (idempotent law), so the expression becomes $A + AC + AB + BC$.

Factor out A: $A(1 + C + B) + BC$. Because $1 + \text{anything} = 1$, this becomes $A \cdot 1 + BC = A + BC$.

This is the distributive law of OR over AND, which has no match in ordinary arithmetic.

8. Answer B $((A' + B') \cdot C')$

Apply De Morgan's law to the OR first: $(AB + C)' = (AB)' \cdot C'$.

Apply De Morgan's law again to $(AB)' = A' + B'$.

The result is $(A' + B') \cdot C'$. Option A forgets that the OR turns into an AND. Option D complements each variable but does not change AB into $A' + B'$.

9. Answer D $(A = 0, B = 1, C = 1)$

Evaluate both product terms for every option:

Option	A B C	$A \cdot B'$	$A' \cdot C$	Y
A	0 0 0	0	0	0
B	1 1 0	0	0	0
C	1 1 1	0	0	0
D	0 1 1	0	1	1

Only option D gives $Y = 1$, because $A' = 1$ and $C = 1$.

10. Answer B (Program Counter)

During the fetch stage, the address in the PC is copied into the MAR, and then the PC is incremented so that it points to the next instruction. The MAR holds whatever address is being accessed at that moment. The IR holds the instruction currently being decoded. The accumulator holds results from the ALU.

11. Answer A (1 MB)

An n-bit address bus can carry 2^n different addresses. $2^{20} = 1,048,576$ addresses.

Each address selects one byte, so the memory is 1,048,576 bytes = 1 MB (with 1 MB = 2^{20} bytes). A 32-bit address bus would give 4 GB (option D).

Memory sizes use binary prefixes: 1 MB is taken as 2^{20} bytes, sometimes written 1 MiB. Data rates and file sizes in networking often use decimal prefixes (1 MB = 10^6 bytes), which is why Q18 states its own convention.

12. Answer A

Data moves both ways: into the CPU on a read and out of it on a write, so the data bus is bidirectional. Addresses are generated by the CPU and sent to memory or I/O, so in the basic model the address bus is one-way. The number of addressable locations depends on the width of the address bus, not the data bus, so D is wrong. The width of the data bus and registers is linked to word size, so B is wrong. The control bus carries signals such as read, write, clock and interrupt, not data values, so C is wrong.

13. Answer D (DRAM)

DRAM stores each bit as charge on a capacitor. The charge must be refreshed constantly, and it

is lost when the power goes off, so DRAM is volatile. ROM, EEPROM and flash memory are all non-volatile.

14. Answer C (17 ms)

Trace the ready queue one time slice at a time (quantum = 4 ms):

Time (ms)	Runs	Time left after the slice
0 – 4	P1	P1: $10 - 4 = 6$
4 – 7	P2	P2: $3 - 3 = 0$ (finishes at 7)
7 – 11	P3	P3: $6 - 4 = 2$
11 – 15	P1	P1: $6 - 4 = 2$
15 – 17	P3	P3: $2 - 2 = 0$ (finishes at 17)
17 – 19	P1	P1: 0 (finishes at 19)

P3 finishes at 17 ms. A process that needs less than a full quantum gives up the CPU as soon as it finishes, as P2 does at 7 ms.

15. Answer D

A compiler, which is a language translator, turns source code into object code. A compiler is system software, but it is a separate program and not part of the operating system. Memory management, process scheduling and file management are core OS functions.

16. Answer C (Router)

A router reads the destination IP address, which is a network-layer (layer 3) address, and chooses the path to another network. A switch forwards frames within one LAN using MAC addresses at layer 2. Hubs and repeaters work at the physical layer and only repeat signals.

17. Answer A (32 bits)

An IPv4 address has 32 bits, written as four decimal octets such as 192.168.10.5. An IPv6 address has 128 bits, and a MAC address has 48 bits.

18. Answer B (4 s)

Step 1: convert the file size to bits. $5 \text{ MB} = 5 \times 10^6 \text{ bytes} \times 8 \text{ bits per byte} = 40 \times 10^6 \text{ bits}$.

Step 2: time = data $\div$ rate = $(40 \times 10^6 \text{ bits}) \div (10 \times 10^6 \text{ bits/s}) = 4 \text{ s}$.

Option A (0.5 s) comes from forgetting to multiply by 8. Link speeds are given in **bits** per second, while file sizes are given in **bytes**.

19. Answer A

A foreign key links two tables. Its values must match primary-key values in the related table, and this is how referential integrity is enforced. Option B describes a primary key (unique and not null).

20. Answer B (2 rows)

A row is returned only if **both** conditions are true.

Name	Marks > 70?	City = 'Lahore'?	Returned?
Ali	Yes (82)	Yes	Yes
Sana	No (68)	Yes	No
Bilal	Yes (75)	No	No
Hira	Yes (91)	Yes	Yes
Usman	No (70 is not greater than 70)	Yes	No

Two rows are returned: Ali and Hira. Usman is the trap, because > excludes 70 and only >= would include it.

21. Answer A (3 2)

Both operands are int, so / does integer division and drops the fraction: $17 / 5 = 3$. The % operator gives the remainder: $17 = 5 \times 3 + 2$, so $17 \% 5 = 2$. The output is "3 2".

22. Answer C (6 7)

$x++$ is post-increment, so the expression uses the old value of x and only then increases x .

$y = 5 + 2 = 7$. After this statement, $x = 6$.

printf prints x first and then y , which gives "6 7". With pre-increment ($++x$), y would be 8.

23. Answer C (7)

* and % have the same precedence, which is higher than +, and they group from left to right.

Step 1: $3 * 4 = 12$.

Step 2: $12 \% 7 = 5$.

Step 3: $2 + 5 = 7$.

Each wrong option comes from a particular mistake with precedence:

- A (0): doing the + before the % gives $(2 + 3 * 4) \% 7 = 14 \% 7 = 0$. This is the mistake of giving % the lowest precedence.
- B (14): working right to left instead of left to right gives $2 + 3 * (4 \% 7) = 2 + 3 * 4 = 14$.
- D (6): working strictly left to right and ignoring precedence gives $(2 + 3) * 4 \% 7 = 20 \% 7 = 6$.

24. Answer B (7)

i takes the values 1, 4, 7, 10, 13, 16, 19. The next value, 22, fails $i \leq 20$ and the loop stops. That is 7 passes through the loop, so count = 7.

A quick formula: number of passes = $\lfloor (20 - 1) \div 3 \rfloor + 1 = 6 + 1 = 7$.

25. Answer C (10)

Each pass adds the last digit ($n \% 10$) and then removes it ($n /= 10$):

Pass	$n \% 10$	s	n after $n /= 10$
1	4	4	123
2	3	7	12
3	2	9	1
4	1	10	0

n is now 0, so the loop ends and the program prints 10, which is the sum of the digits.

26. Answer D (BC)

k = 2, so control jumps to case 2 and prints B. case 2 has no break, so execution **falls through** into case 3 and prints C. The break then leaves the switch, so default is never reached. The output is "BC".

27. Answer A (11)

i goes 0, 2, 4 (the next value, 6, fails $i < 5$), so the loop adds the elements at the even indices.

$a[0] + a[2] + a[4] = 4 + 2 + 5 = 11$.

Array indices start at 0, so $a[1]$ is 9, not 4. Adding the odd-index elements gives 16 (option B), and adding the whole array gives 27 (option C).

28. Answer C (4)

C passes arguments **by value**. The parameter x gets a copy of n, and the function changes only that copy (x becomes 40). The copy is discarded when the function returns, so n in main is still 4. Changing n would need a pointer (pass by address) or a return value.

29. Answer D (10)

For each value of i, the inner loop runs i times:

i = 1: 1 star; i = 2: 2 stars; i = 3: 3 stars; i = 4: 4 stars.

Total = $1 + 2 + 3 + 4 = 10$. The general formula is $n(n + 1) \div 2$ with $n = 4$, which gives $4 \times 5 \div 2 = 10$. Option C ($16 = 4 \times 4$) would be right only if the inner loop ran up to 4 every time.

30. Answer B (3.0)

The right-hand side $7 / 2$ is worked out first. Both operands are int, so the result is the integer 3. Only then is it converted to float for the assignment, so $f = 3.0$. The format `%.1f` prints one decimal place, which gives "3.0".

Option D (3) is wrong because `%.1f` always prints exactly one digit after the decimal point, even

when that digit is 0.

To get 3.5, at least one operand must be a floating-point value, for example $7.0 / 2$ or $(\text{float})7 / 2$.

Common mistakes this paper tests

- Forgetting the "+1" step in two's complement (Q3).
- Mixing up bits and bytes in transfer-time questions (Q18).
- Treating $>$ as if it were $\geq$ in conditions (Q20).
- Miscounting loop passes at the boundary of a $\leq$ condition (Q24).
- Applying operators in the wrong order: $*$, $/$ and $\%$ come before $+$ and $-$, and work left to right (Q23).
- Integer division in C, including when the result is assigned to a float (Q21, Q30).
- Post-increment versus pre-increment (Q22).
- Missing break statements in a switch (Q26).
- Expecting a function to change a variable that was passed by value (Q28).