紙筆測驗 Paper-Based

視 覺 障 礙 者 請 事 先 告 知
Let me know in advance
if you are visually impaired

考試時間 Test time 15:51 pm to 18:18 pm. 以教室後方時鐘為準 Using the clock at the back of the classroom as official time. 在**答案卷**上作答 Answer on the **answer sheet**. 每題一分 One point per problem. 全對才給分 No partial credit. 不倒扣 No penalty for wrong answers. 禁止外部資源（比照學測、指考） No external resources.

對的選項可以按 Correct choices are clickable. 請小心負面表列的問題 Be careful with negatively phrased questions. 題目很多，請合理分配時間 There are too many questions, so manage your time wisely. 搞清楚錯的選項為什麼錯很重要 It is important to understand why wrong choices are wrong. 因為選擇題可能會升級成填充題 Because multiple choice questions may be upgraded to fill-in-the-blank questions.

Answer Sheet

Name (zh or en) Hsin-Po Wang　　Student ID B00201054

[0] B　　[1] I　　[2] L

（反正就像這樣啦，後面的題號不想做了）

1 Basics

[0] The code to the right outputs (A) `hollywood` (B) `helloworld` (C) `hello world` (D) `Hello World` (E) `hello<<world`

```
#include <iostream>
using namespace std;
int main() {
cout << "hello"
     << "world";
}
```

[1] The purpose of a `hello world` program is to (F) demonstrate how to import data (G) benchmark the performance of the computer it runs on (H) greet the world as the artificial intelligence wakes up (I) test if compiler, as well as the environment, is installed correctly

[2] For a beginner who just finished the `hello world` program, a good next step is to write (J) a compiler that can compile C programs (K) a `hello world` program in another language (L) a program that takes two numbers and outputs their sum (M) a meta-program that outputs a program that outputs `hello world` (N) a program that can be compiled by C, Perl, Python, and Ruby compilers at the same time

See also 1.

See also 2.

[3] `iostream` is a (O) general-purpose strongly-typed programming language (P) random variable that obeys the standard normal distribution (Q) function that takes no input and returns a stream as output (R) header file that provides basic input and output functionalities

[4] `using namespace std;` is to (S) include the standard libraries (T) define `namespace` to be the new standard (U) avoid typing `std::` before commonly-used functions (V) enforce the naming standard that spaces are allowed in variable names

[5] Which one is incorrect about `main`? (W) It marks the entry point of a program; everything starts here. (X) `argc` and `argv` take arguments from the command line; they are optional; `int main()` is perfectly fine. (Y) If we have nothing to return, using `void* main()` is acceptable and using `double** main()` is recommended over `int main()`. (Z) Conventionally, returning `0` means that the program exits without an error. Omitting `return` is perfectly fine and is equivalent to `return 0`.

```
int main (int argc, char *argv[]) {
    return 0;
}
```

[6] Which one is incorrect regarding this compiling command? (A) `-o op.exe` specifies the output file name. (B) `-Wall` turns on some warnings. Reducing the number of warnings is a good practice. (C) `gcc` refers to the GNU Compiler Collection, other compilers include `clang`, `icc`, `nvcc`, etc. (D) `-Ofast` is an optimization flag; it makes the program run faster but may produce incorrect results. (E) `option_pricing.c`, the input file name, suggests that it is not a C++ file, so `g++` should be used instead of `gcc`.

```
gcc -Ofast -Wall -o op.exe option_pricing.c
```

[7] Why do C/C++ programmers pay extra attention to undefined behavior? (F) Because it will definitely crash the program. (G) Because it will definitely crash the compiler. (H) Because codes containing undefined behavior may work on my machine but not on yours. (I) Undefined behaviors make it impossible for eavesdropper to steal data from the program.

2 Variables

[8] Which one is a valid variable name? (J) `finalproject.pptx` (K) `final project (2)` (L) `__Final_Project__` (M) `final-project-latest` (N) `finalproject2024/10/24`

[9] Which one is the most informative variable name? (O) `num` (P) `SID` (Q) `std::ID` (R) `studentIDnum`

[10] `cout << int('a') << int('b') << int('c');` **outputs 979899.**

Accordingly, `cout << int('x') << int('y') << int('z');` **outputs.....**

see also

[11] `int a = 3.14; cout << a;` **prints .....**

[12] Sometimes, `double` is preferred over `float` because (S) `double` runs twice as fast as `float` (T) `double` contains nan but `float` does not (U) `double` is a built-in type and `float` is not (V) `double` has more precision and range than `float` (W) we need to store an even number instead of a real number

[13] `int` is an integer type of (X) 8-bits (Y) 16-bits (Z) at least 1-bit (A) at least 16-bits (B) at most 116-bits

[14] The 2038 Problem refers to (C) The IPv4 address space being exhausted in 2038 (D) the world ending in 2038 as predicted by Maya calendar (E) the representation of year by the last two digits overflowing in 2038 (F) the representation of time by seconds since 1970-01-01 overflowing in 2038

3 Arithmetic

After the code to the right is executed,

```
int a = -2, b = 5, c = 2;
a += b; b -= c; c *= a;
```

[15] `a` becomes .....

[16] `b` becomes .....

[17] `c` becomes .....

After the third line being executed,

```
a /= b; b %= c; c != a;
```

[18] `a` becomes .....

[19] `b` becomes .....

[20] `c` becomes .....

After the fourth line being executed,

```
a >= b; b <= c; c == a;
```

[21] `a` becomes .....

[22] `b` becomes .....

[23] `c` becomes .....

After the fifth line being executed,

```
a >>= b; b <<= c; c =  a;
```

[24] `a` becomes .....

[25] `b` becomes .....

[26] `c` becomes .....

After the sixth line being executed,

```
a |= b; b ^= c; c &= a;
```

[27] `a` becomes .....

[28] `b` becomes .....

[29] `c` becomes .....

[30] For an integer `a` between $1,000,000$ and $999,999,999$, we can print `a` in the format `123,456,789`

with `cout <<` .....`;`

[31] Using `a`, `b`, `c`, `x`, `add`, and `mul`, instead of `+` and `*`, one can express quadratic polynomial

$ax^2 + bx + c$ as .....

[32] BMI, the outdated body mass index, is the quotient of weight in kilograms and height in meters squared. Complete the following code.

```
cout << "How tall are you in centimeters?";
cin >> height;
```

```
cout << "How heavy are you in kilograms?";
cin >> weight;
cout << "Your BMI is " <<
;
```

[33] Why it is a bad idea to write `d = c++ + c++ + ++c;`? (G) It is not a bad idea; it compiles and runs perfectly fine. (H) Because it is unclear how many times `c` should increase before and after addition. (I) Because there is an internet subculture that recommends avoiding the name of the language in codes of that language. (J) Because three `+` in a row stand for the `superplus` operator, and five `+` in a row stand for the `ultraplus` operator.

4 If and For

[34] Why some people prefer the Yoda notation `if(42==y)` over `if (y == 42)`? (K) Because it takes three fewer characters to type. (L) Because C evaluates function arguments from right to left, so `y` will be evaluated first. (M) Because `if(42=y)` is a syntax error, so it is not possible to make the mistake `if(y=42)`, which is always true.

[35] What seems to be the problem with the code to the right? (N) Functions must to be named in CamelCase, so `sSHaUTHENTIFICATION` is better. (O) There is a semicolon after the `if` statement, so `goto fail` is always executed. (P) If an equal sign is accidentally deleted, `SSHAuthentification() = 0` will redefine the function. (Q) `SSHAuthentification` probably returns a boolean value, and it is illegal to compare it to an integer.

```
if (SSHAuthentification() == 0);
    goto fail;
```

See also.

[36] In the famous article Go To Statement Considered Harmful, Dijkstra argues that (R) `goto` is a keyword in C and should not be used in C++. (S) `goto fail` reminds people of "goto hell", which is offensive (T) jumping from line to line makes the program harder to understand, so it should be avoided. (U) `goto` skips lines and make the program run faster, so cloud providers cannot charge user by the hour.

[37] When the loop ends, `sum` is .....

[38] When the loop ends, `j` is (V) 0 (W) 9 (X) 10 (Y) 11 (Z) not usable, as `j` is local to the loop

```
int sum = 0;
for (int j = 0; j < 10; j++) {sum += j;}
```

[39] What does the following code do? (Your description should be detailed enough such that modern AI tools can recover the code from it.)

```
int guess = 0;
int target = rand() % 100 + 1;
while (guess != target) {
    cin >> guess
    if (guess > target) { cout << "Too high"; }
    else if (guess < target) { cout << "Too low"; }
    else { cout << "Good "; }
}
```

[40] Write a function `HashBrown` such that `HashBrown(3)` couts the pattern to the right, and `HashBrown(n)` for general `n` couts a similar pattern with side length `n`.

```
  #
 ###
#####
 ###
  #
```

```
void HashBrown(int n) {
```

```
}
```

5 Arrays

[41] Which one creates an array? (A) `A = [1, 2, 3];` (B) `A = 1, 2, 3;` (C) `int array[3];` (D) `A = array(length=3);` (E) `A = new (*float) <3>;`

[42] Declared as `int A[100];` the type of `A` is best described as (F) an array (G) an integer array (H) an array of length 100 (I) an array of 100 integers

[43] Given `int A[100];` trying to access `A[135]` results in (J) `A[35]` (K) `A[99]` (L) undefined behavior (M) `A` automatically resizing to 135

[44] Given `int A[100];` trying to access `A[-70]` results in (N) `A[0]` (O) `A[30]` (P) undefined behavior (Q) the last 70 elements of `A` being dropped

[45] Which one sums all elements of `A`? (R)
```
for(i = 1; i <= 100; i++)
    sum += A[i];
```
(S)
```
for(i = 99; i >= 0; --i)
    sum += A[i];
```
(T)
```
for(i = 1; i < 100; ++i)
    sum =+ A[i];
```

[46] Which one finds the maximum element of `A`? (U)
```
max1 = 0;
for(i = 0; i < 100; i++) {
    if(A[i] > max1) {max1 = A[i];}
}
```
(V)
```
max1 = A[0];
for(i = 1; i < 100; i++) {
    if(A[i] > max1) {max1 = A[i];}
}
```
(W)
```
max1 = 0;
for(i = 0; i < 100; i++) {
    max1 = max(A[i], max1);
}
```

[47] Joseph wants to solve heat equation $\frac{du}{dt} = \frac{d^2u}{dx^2} + \frac{d^2u}{dy^2}$ using finite element method. What seems to be wrong with his code?

```
void EvolveOneSecond(float Temperature[100][100], float conductivity) {
    float Temporary[100][100];
    for (int x = 0; x < 100; x++) {
        for (int y = 0; y < 100; y++) {
            float ThisT = Temperature[x][y];
            float SumDiff = 0;
            SumDiff += Temperature[x-1][y] - ThisT;
            SumDiff += Temperature[x+1][y] - ThisT;
            SumDiff += Temperature[x][y-1] - ThisT;
            SumDiff += Temperature[x][y+1] - ThisT;
            Temporary[x][y] = SumDiff * conductivity;
        }
    }
    for (int x = 0; x < 100; x++) {
        for (int y = 0; y < 100; y++) {
            Temperature[x][y] = Temporary[x][y];
        }
    }
}
```

6 Functions

[48] What defines a function? (X) `lambda a & b? a : b;` (Y) `#define add(a, b) := a + b;` (Z) `new add(a, b) {return a + b;}` (A) `int mul(int a, int b) {return a + b;}`

[49] Which one is the most informative function name? (B) `Load()` (C) `Print()` (D) `Initialize()` (E) `RemoveNanAndZero()`

[50] Which one of the function definitions matches its name? (F) `int Mod2(int a) {return a//2;}` (G) `int Power(int a, int b) {return a^b;}` (H) `int Sum(int a, int b) {return a + b;}` (I) `int Atan2(int a, int b) {return sin(a) / cos(b);}`

[51] Select all functions that are commented out. (J) `LoadTable` (K) `CheckLoaded` (L) `RemoveNan` (M) `RemoveZero` (N) `SumColumns` (O) `PrintSums` (P) `Dump` (Q) `CheckAvailableSpace`

```
// LoadTable();
   CheckLoaded();
// RemoveNaN(); // RemoveZero();
/* SumColumns(); */
/* PrintSums();
 * Dump();
 */CheckAvailableSpace();
```

[52] Why is there a star before `Dump()`? (R) To multiply `Dump()` by `PrintSums()`. (S) It is part of syntax of multiline comments. (T) `Dump` is a pointer to the actual dumping function. (U) `Dump()` returns a pointer and we want to dereference it. (V) Aligning three stars in a row pays respect to the Michelin Guides. (W) By prefixing every line with a star, even if the comments are so long that `/*` and `*/` are not on the screen, it is still clear that they are comments.

[53] What makes a function recursive? (X) It causes an infinite loop. (Y) It appears in the code many times. (Z) It calls itself with different arguments. (A) It remains in the codebase after a major update.

7 Pointers and Memory

[54] The code to the right outputs (B) `10` (C) `20` (D) `30`

```
int a = 10;
int *p = &a;
*p += 20;
cout << a;
```

[55] Why is `malloc(100 * sizeof(int))` preferred over `malloc(400)`? (E) Because `int` may be 4 bytes on some systems but 8 bytes on others. (F) Because `sizeof(int)` is 2, so `400` is twice of what is actually needed. (G) Because `malloc` will return an integer pointer if we specify `int` explicitly. (H) Because the compiler runs faster if it does not have to evaluate `100 * sizeof(int)`.

After the code to the right is executed,

```
int a = 7;
int b = 11;
int *p1 = &a;
int *p2 = &b;
*p1 += *p2;
p2 = p1;
p1 = p2;
*p2 += 13;
```

[56] `a` is .....

[57] `b` is .....

[58] `p1` points to .....

[59] `p2` points to .....

[60] Knowing that `[]` takes precedence over `*`, the declaration `int **a[10][20][30];` makes `a` a

.....

8 Structures

[61] `struct { int a; float b; }` is a structure consisting of an integer and a float. `struct Node` is a structure consisting of

a .....,

```
structure Node {
    double a;
    Node *b;
    Node *c;
}
```

a ......, and

a .....

[62] Recall that $i = \sqrt{-1}$. Assume that I use the code to the right to print complex numbers. Guess the structure of `Cmplx`.

```
string Cmplx2Strng (struct Cmplx z) {
    string answer = to_string(z.a);
    answer += " + ";
    answer += to_string(z.b);
    answer += "i";
    return answer;
}
```

`struct Cmplx {`

`};`

[63] Implement addition of complex numbers.

`struct Cmplx add(struct Cmplx x, struct Cmplx y) {`

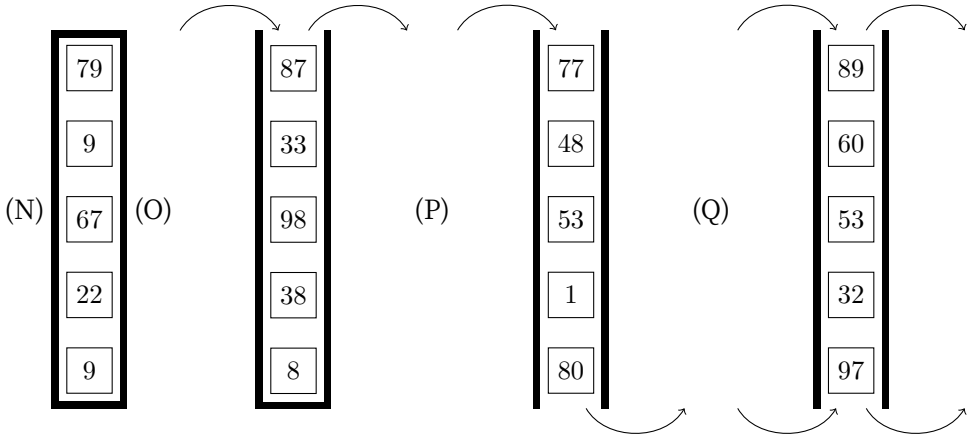
`}`

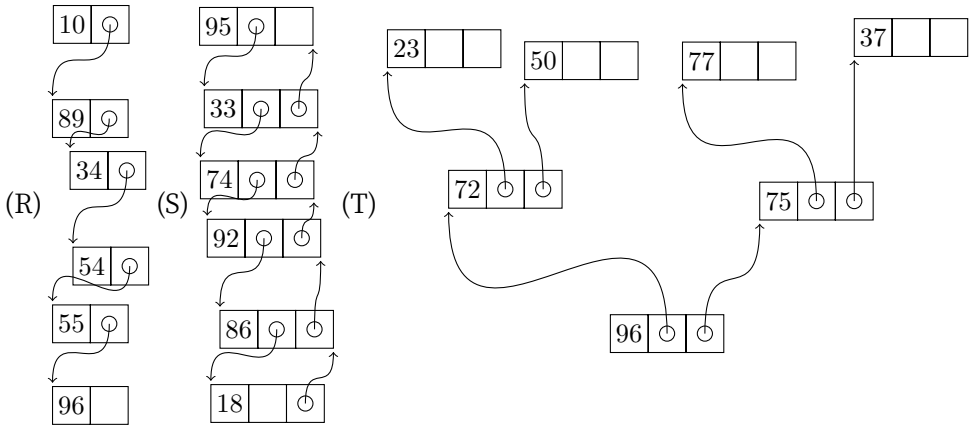
[64] Implement multiplication of complex numbers.

`struct Cmplx mul(struct Cmplx x, struct Cmplx y) {`

`}`

[65] When a stack is empty, what happens if we try to pop? (I) The compiler will err. (J) The stack will overflow. (K) It causes a time limit exceed. (L) It will wait for the next push and immediately pop it. (M) It may return NaN or the program may crash, depending on the implementation.





[66] Which drawing represents a fixed-length array?

[67] Which drawing represents a stack (LIFO)?

[68] Which drawing represents a queue (FIFO)?

[69] Which drawing represents a double-ended queue?

[70] Which drawing represents a (singly) linked list?

[71] Which drawing represents a doubly linked list?

[72] Which drawing represents a binary tree?

9 Algorithm

[73] [LeetCode560] `Material` is a 1D array of n positive integers and `target` is an integer. Your goal is to `cout <<` all pairs of indices (i, j), one pair per line, such that `Material[i] + Material[i+1] + ... + Material[j] == target`. As a non-mandatory hint, you may use `PrefixSum` that is defined as `PrefixSum[i] = Material[0] + Material[1] + ... + Material[i]`.

```
void WeightMatching(int Material[], int n, int target) {
```

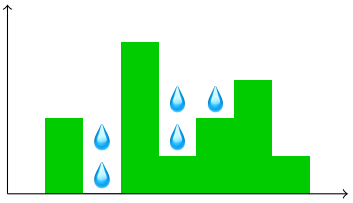
```
}
```

[74] Same as the previous problem, but your algorithm should run in $O(n)$ time. Getting this problem right will grant you point for the previous problem. As a non-mandatory hint, recall that integers in **`Material`** are positive and think about how to implement a queue using an array and two indices/pointers?

```
void WeightMatching(int Material[], int n, int target) {

}
```

[75] [LeetCode42] **`Elevation`** is a 1D array of n nonnegative integers. An example is **`int Elevation[7] = {0, 2, 0, 4, 1, 2, 3, 1};`** as shown to the right by the green blocks. Now it rains, very heavily, and the water is trapped, as represented by the blue!75!black droplet emoji. Write a function that computes the area of trapped water.



```
int TrapWater(int Elevation[], int n) {

}
```

[76] Same as the previous problem, but your algorithm should run in $O(mn)$ time and $O(m+n)$ space, where n is the length of **`Elevation`** and m is the maximum elevation. Getting this problem right will grant you point for the previous problem. As a non-mandatory hint, you may iterate from $x = 0$ to n and maintain an array **`int Cliff[m];`** such that **`Cliff[y]`** is the last x coordinate where there is an east-facing cliff at height y.

```
int TrapWater(int Elevation[], int n, int m) {

}
```

Good Choice 👍

Correct choices are made into hyperlinks. They link you to this page. Next time, hover your mouse over choices to see if they are correct.

上機測驗 Computer-Based

考試時間 Test time 15:51 pm to 17:17 pm. 以 NTU COOL 時鐘為準 Using the clock of NTU COOL as official time. 上傳 cpp 檔到**作業區** Upload cpp files to the **assignment section**. 每個題號代表一筆測資 Every problem number represents one testing case. 每筆測資一分 One points per testing case. 可上網及查書 You may loop up information online or in books. 可用人工智慧 AI is allowed. 禁止合作討論 No collaboration or discussion. 禁止求助真人、論壇、社群媒體等 Do not seek help from forums, social media, chatting apps, etc.

[77] [78] [79] [80] Write a function that takes an array `char Image[100][100][3];` as input and outputs nothing. The input is treated as a picture where `Image[x][y]` is the color of the pixel at (x, y). Rotate this image in-place by 90 degrees counterclockwise.

```
void RotateImage (char Image[100][100][3]) { ... }
```

[81] [82] [83] [84] [85] [86] On Jan 5, 2030, you borrowed L TWD, $1 \leq L \leq 10^{12}$ from a bank to invest in chip manufacturing. On the 10th day of each month, the bank adds a 0.2% interest to your outstanding balance, rounded up to the nearest 0.1 TWD. On the 15th day of each month, you pay the bank P TWD. You wish to repay the loan by Dec 20, 2040 while minimizing your monthly payment. Write a function that takes `long long L` as input and returns the least possible `long long P`.

```
long long MinimizeMonthly (long long L) { ... }
```

Binary search (or any sub-linear algorithm) is necessary for the last two testing case.

[87] [88] [89] [90] [91] [92] [93] [94] [95] [96] [97] [98] [99] [100] [101] [102] [103] [104] Write a function whose input is a string like `1+(2-3)*4/5^6` (whose length ≤ 100) and outputs

```
1+(2-3)*4/5^6
   \_/
1+( 6 )*4/5^6
1+  6  *4/5^6
         \_/
1+  6  *4/ 1
       \__/
1+  6  * 4
    \____/
1+     3
\_______/
 4
```

Read the following rule very carefully.

- **All arithmetic are modulo 7. Only digits $0, 1, 2, 3, 4, 5, 6$ are allowed.**
- **In the first three test cases, only `+` will appear. Evaluate from left to right. Example: `1 + 1 + 1` becomes `2 + 1`.**
- **`+` is always a binary operator, never unary.**
- **Starting from the fourth test case, `-` will appear. Evaluate it before any evaluation of `+` and from left to right. Example: `1 + 2 - 3 - 4` becomes `1 + 6 - 4` because $-1 \equiv 6 \pmod 7$.**
- **`-` is always a binary operator, never unary.**
- **Starting from the seventh test case, `*` may appear. Evaluate `*` before any evaluation of `+` and `-` and from left to right. Example: `5 * 6 * 0` becomes `2 * 0` because $30 \equiv 2 \pmod 7$**
- **Starting from the tenth test case, `/` may appear. Evaluate `/` before any evaluation of `+`, `-`, and `*` and from left to right. Example: `1 / 2 / 3` becomes `4 / 3` because the solution to $1 \equiv 2x \pmod 7$ is $x \equiv 4 \pmod 7$.**
- **Starting from the thirteenth test case, `^` will appear and mean power. Evaluate `^` before any evaluation of `+`, `-`, `*`, and `/` and from left to right. Example: `4 ^ 5 ^ 6` becomes `2 ^ 6` because $4^5 \equiv 2 \pmod 7$. The exponent can be $1, 2, 3, 4, 5, 6$; it can never be 0;**

- Starting from the sixteenth test case, parenthesis will appear. Evaluate the leftmost parentheses before everything else.
- When evaluating, put backslash under the left operand; put slash under the right operand; put underscores between backslash and slash; put the result below the operator.

```
void TeachArithemteic (string Expression) { ... }
```