1EE2310 C++程式設計 HW2 (Ch 4,5)

**Part 1. Choice選擇題:30%**

1) Relational operators allow you to \_\_\_\_\_\_\_\_ numbers.

A) add

B) multiply

C) compare

D) average

E) verify

2) The \_\_\_\_\_\_\_\_ statement can cause other program statements to execute only under certain conditions.

A) logical

B) relational

C) cin

D) cout

E) if

3) In C++ when a relational expression is false, it has the value \_\_\_\_\_\_\_\_.

A) 1

B) 0

C) -1

D) "0"

E) of any negative number

4) A(n) \_\_\_\_\_\_\_\_ is a variable, usually a bool, that signals when a condition exists.

A) flag

B) identifier

C) named constant

D) condition variable

E) logical variable

5) The \_\_\_\_\_\_\_\_ statement executes one block of statements if a test condition is true, and another block if the condition is false.

A) if

B) if/else

C) if/else if

D) switch

E) trailing else

6) The \_\_\_\_\_\_\_\_ statement executes one block of statements if a test condition is true, and another block if the condition is false.

A) if

B) if/else

C) if/else if

D) switch

E) trailing else

7) A trailing else placed at the end of an if/else if statement provides a default action when \_\_\_\_\_\_\_\_ of the if conditions is/are true.

A) none

B) any one

C) only the last one

D) at least two

E) all

8) When an if statement is placed within the conditionally-executed code of another if statement, this is known as a(n) \_\_\_\_\_\_\_\_.

A) complex if

B) overloaded if

C) nested if

D) conditional if

E) double if

9) The C++ \_\_\_\_\_\_\_\_ operator represents logical AND.

A) ++

B) |

C) ||

D) &

E) &&

10) The \_\_\_\_\_\_\_\_ operator takes an operand and reverses its truth or falsehood.

A) relational

B) &&

C) ||

D) !

E) !=  
11) The \_\_\_\_\_\_\_\_ statement acts like a chain of if statements. Each performs its test, one after the other, until one of them is found to be true.

A) if/then

B) if/else

C) if/elseif

D) if/notif

E) if/endif

12) A flag is a variable, usually of data type \_\_\_\_\_\_\_\_, that signals when a condition exists.

A) char

B) double

C) long

D) bool

E) logical

13) When a program lets the user know that an invalid menu choice has been made, this is an example of \_\_\_\_\_\_\_\_.

A) input validation驗證

B) output validation

C) menu reselection

D) invalidation

E) being user unfriendly

14) If s1 and s2 are string objects, s1 == s2 is true when \_\_\_\_\_\_\_\_.

A) s1 = "lion" and s2 = "lioness"

B) s1 = "dog" and s2 = "DOG"

C) s1 = "cat" and s2 = "cat "

D) None of these because one or more characters in the strings have different ASCII codes.

E) None of these because string objects cannot be compared with relational operators.

15) The default section of a switch statement performs a similar task as the \_\_\_\_\_\_\_\_ portion of an if/elseif statement.

A) conditional test

B) break

C) trailing else尾端的

D) elseif

E) body

16) \_\_\_\_\_\_\_\_ are C++ operators that change their operands by one.

A) + and -

B) ++ and --

C) binary and unary

D) arithmetic and relational

E) conditional and relational

17) The ++ operator \_\_\_\_\_\_\_\_.

A) is a unary operator

B) adds one to the value of its operand

C) can operate in prefix or postfix mode

D) All of the above.

E) Both B and C, but not A.

18) The while loop has two important parts: a condition that is tested and a statement or block of statements that is \_\_\_\_\_\_\_\_.

A) repeated as long as the condition is true

B) repeated until the condition becomes true

C) done once if the condition is true

D) always done at least once, then repeated if the condition is true

E) always skipped

19) The while loop is a(n) \_\_\_\_\_\_\_\_ loop and the do-while loop is a(n) \_\_\_\_\_\_\_\_ loop.

A) finite, infinite

B) infinite, finite

C) simple, complex

D) pretest, post test

E) post test, pretest

20) The statements in the body of a do-while loop are always executed \_\_\_\_\_\_\_\_.

A) exactly once

B) at least once

C) at least twice

D) forever until the user hits the break key

E) until the test condition becomes true

21) A(n) \_\_\_\_\_\_\_\_ is a variable that controls the number of times a loop iterates.

A) counter

B) accumulator

C) sentinel

D) total

E) loop control variable

22) A(n) \_\_\_\_\_\_\_\_ is a variable that is regularly incremented or decremented each time a loop iterates.

A) counter ／／ｉ＋＋

B) accumulator

C) sentinel

D) total

E) loop control variable

23) A(n) \_\_\_\_\_\_\_\_ is a special value that marks the end of a list of values.

A) counter

B) accumulator

C) sentinel

D) total

E) loop control variable

24) The statements in the body of a while loop will never be executed if the test condition \_\_\_\_\_\_\_\_.

A) is initially true

B) is initially false

C) changes from true to false

D) changes from false to true

E) is not Boolean

25) The do-while loop is a(n) \_\_\_\_\_\_\_\_ loop and the while loop is a(n) \_\_\_\_\_\_\_\_ loop.

A) finite, infinite

B) infinite, finite

C) simple, complex

D) pretest, post test

E) post test, pretest

26) In a for statement, the \_\_\_\_\_\_\_\_ expression is executed only once.

A) initialization

B) test

C) repeat

D) validate

E) update

27) The \_\_\_\_\_\_\_\_ statement causes a loop to terminate early.

A) stop

B) break

C) quit

D) terminate

E) continue

28) If a while loop has no braces around the body of the loop \_\_\_\_\_\_\_\_.

A) there is no loop body

B) the loop body ends when the endwhile statement is encountered

C) the loop body contains one statement

D) the loop body contains one line

E) a compiler error will occur

29) The ideal type of loop to use if you want a user to enter exactly 20 values is a(n) \_\_\_\_\_\_\_\_ loop.

A) do-while

B) for

C) sentinel controlled

D) infinite

E) nested

30) The ideal type of loop to use for repeating a menu is a(n) \_\_\_\_\_\_\_\_ loop.

A) do-while

B) for

C) sentinel controlled

D) infinite

E) nested

**Part 2. 填充題:10%**

1) True/False: Relational expressions and logical expressions are both Boolean, which means they evaluate to true or false.

2) True/False: In C++ an expression that evaluates to 5, -5, or for that matter anything other than 0, is considered true by an if statement.

3) True/False: Assuming moreData is a Boolean variable, the following two tests are logically equivalent.

if (moreData == true)

if (moreData)

4) True/False: The block of code in the body of a while statement can contain an unlimited number of statements, provided they are enclosed in braces.

5) True/False: An initialization expression may be omitted from the for loop if no initialization is required.

6) True/False: The for statement has three expressions in its form:

for (expr1;expr2;expr3)

Any or all of the expressions can be omitted in the for statement.

7) True/False: To exit from a nested loop, a break; statement in the innermost loop is needed.

8) True/False: Generally, the for loop can be implemented using a while loop. Therefore, the following two code segments are totally identical:

int i,n=0,sum=0; int i,n=0,sum=0;

while(n<10){ for(n=0;n<10;n++){

cin >> i; cin >> i;

if(i==0) continue; if(i==0)continue;

sum += i; n++; sum+=i;

} }

9) True/False: The output produced by the following for statement would be a kind of count-down sequence starting from 10 down to 0 in some finite steps.

for(int i=10;i>=1;i/=2) cout << i++ << “ “;

10) True/False: The following two statements are equivalent in every way. (assuming that the loop bodies are the same)

(a) for(i=0; i<10; i++) {…}　１０

(b) for(i=0; i++<10;) {…}　ｉ＝１１

**Part 3. Programming 程式題:60%**

**1. Internet Service Provider Part 1**

An International Internet phone company has three different subscription packages for its customers:

Package A: For $9.95 per month 5 hours of call time are provided. Additional usage costs $0.08 per minute.

Package B: For $14.95 per month 10 hours of call time are provided. Additional usage costs $0.06 per minute.

Package C: For $19.95 per month unlimited call time is provided.

Write a program that calculates a customer’s monthly bill. It should input customer name, which package the customer has purchased, and how many hours were used. It should then create a bill that includes the input information and the total amount due. Wherever possible, use named constants instead of numbers.

Input Validation: Be sure the user only selects package A, B, or C.

**2. Internet Service Provider Part 2**

Modify the program in problem 1 so it also displays how much money Package A customers would save if they purchased packages B or C, and how much money package B customers would save if they purchased package C. If there would be no savings, no message should be printed.

**3. Math Tutor Version 1**

Write a program that can be used as a math tutor for a young student. The program should first display a message “Start practice of adding two numbers:”, then display two random numbers between 10 and 50 that are to be added, such as:

24

+ 12

——

The program should then wait for the student to enter the answer. If the answer is correct, a message of congratulations should be printed. If the answer is incorrect, a message should be printed showing the correct answer. Then the program would ask the student to continue the practice all over again or not (Y/N). If not, quit the program.

**4. Math Tutor Version 2**

Starting with the version described in problem 3, modify the program again so it displays a menu allowing the user to select an addition, subtraction, or multiplication problem. The final selection on the menu should let the user quit the program. After the user has finished the math problem, the program should display the menu again. This process must repeat until the user chooses to quit the program. If the user selects an item not on the menu, the program should print an error message and then display the menu again.

**5. Diamond Display**

Write a program that uses nested loops to display the diamond pattern shown below.

+

+++

+++++

+++++++

+++++

+++

+

**6. Compute mathematical constant *e***

The value of the mathematical constant ***e*** can be expressed as an infinite series:

***e*** = 1 + 1/1! + 1/2! + 1/3! + 1/4! + …..

In numeric computation, it is usually approximated by the following finite series:

***e*** = 1 + 1/1! + 1/2! + 1/3! + 1/4! + ….. + 1/n!

Write a program that approximates ***e*** by continuously adding terms (1/n!) until the current added term becomes less than ****, where **** is a small (floating-point) number entered by the user, e.g., 0.0001.