115CI02

國立臺北科技大學 115 學年度碩士班招生考試

系所組別：2300 資訊工程系碩士班

第二節　程式設計　試題

注意事項：

1. 本試題共 五 題，共 100 分。
2. 不必抄題，作答時請將試題題號及答案依照順序寫在答案卷上。
3. 全部答案均須在答案卷之答案欄內作答，否則不予計分。

Problem 1 [33%] [1-1 ~ 1-9 each 2%, 1-10 ~ 1-14 each 3%]

(1) The following Python programs are generated using a generative AI/LLM. Which of the following is a possible runtime problem when these programs are executed?

For each problem, choose one from: (A) AttributeError (B) Compilation error (C) IndexError: out of bound (D) Logic Error (E)NameError (F) NullPointerException (G) TypeError: not a integer (H) Unable to import library (I) UnboundLocalError: cannot access local variable (K) ValueError: invalid for int() (M) ZeroDivisionError: integer modulo by zero (P) ZeroDivisionError: integer division by zero (Q) Infinite loop execution (R) Unspecified runtime exception (S) Return value is None unexpectedly

```
01  def fA(n: int):                                  # Problem 1-1, fA(0)
02      data = [i for i in range(n) if i < n]
03      index = sum([i for i in range(n)]) - 1
04      return data[index]
05
06  def fB(n: int):                                  # Problem 1-2, fB(1)
07      data = [i for i in range(n) if i < n]
08      index = sum([i for i in range(data)])
09      return data[index]
10
11  def fC(data: str):                               # Problem 1-3, fC('2ac')
12      parts = [d for d in data]
13      idx = len(parts) // 2
14      if idx<0: return 0
15      return int(parts[idx])
16
17  def fD(n):                                       # Problem 1-4, fD(2)
18      data = [i for i in range(n) if i%2==0]
19      if len(data) >= 0:
20          index = n% sum(data)
21          return index
22
23  def fE(data):                                    # Problem 1-5, fE([1, 2])
24      flag = sum(data) == 0
25      if flag: result = 10
26      return result
27
28  def fF(i, n):                                    # Problem 1-6, fF(3, 2)
29      while(i>n): i+=3
30      return i
```

(2) Please trace the following Python program and answer the output of each "print" statement for problems 1-7 ~ 1-14.

```
01  def permu(data: list, n: int, got: int=0, result: list=[])->list:
02      if got == n:
03          result.append(data[:n])
04          return result
05      for i in range(got, len(data)):
06          data[i], data[got] = data[got], data[i]
07          permu(data, n, got + 1, result)
08          data[i], data[got] = data[got], data[i]
09      return result
10
11  def inc_loop(n: int)->int:
12      result = i = 0
13      inc = lambda m, k: m + k
14      while i < n:
15          result += i
16          i = inc(i, i + 1)
17      return result
18
19  def getRank(score: int):
20      rank = lambda mark: print(mark)
21      grade = {10:'A', 9:'A', 8:'B', 7:'C'}.get(score//10, 'F')
22      rank(grade)
23  def add_four(v1: int, v2: int, v3: int = 1, v4 = 5)->int:
24      while True:
25          v2 = v1 + v2
26          v1 = v1 + v2
27          v3 = v1 + v2 + v3
28          if (v1>=v4): break
29      return v3
30  def fib(n: int, r: list):
31      if n==0 or n==1: return n
32      r[0] += 1
33      return fib(n-1, r)+fib(n-2, r)
34  def fibs(data: list, n: int, r: list):
35      if n<=1: data[n] = n
36      elif data[n]==0:
37          data[n] = fibs(data, n-1, r) + fibs(data, n-2, r)
38          r[0] += 1
39      return data[n]
40  def getData(m: int, n: int, index: int, num: int) -> int:
41      data, bias = [0] * num*2, [0] * num*2
42      for i in range(num): data[i] = (n + i) % m + 1
43      for i in range(num): bias[i] = (m + i) % n + 1
44      for i in range(num): data[i] = index * data[i] + bias[i]
45      return data[index]
46  m, n, num, record1, record2 = 5, 6, 7, [0], [0]
47  data_f = [0 for i in range(num+1)]
48  data_r = [[0 for _ in range(m+n)] for _ in range(m+n)]
49  getRank(80)                                      # Problem 1-7
50  print(fib(num, record1)-fibs(data_f, num, record2))   # Problem 1-8
51  print(permu([2, 3],    2, 0, result=[])[1][1])       # Problem 1-9
52  print(permu([3, 4, 5], 3, 0, result=[])[0][2])       # Problem 1-10
53  print(inc_loop(10))                              # Problem 1-11
54  print(getData(8, 6, 1, 5))                       # Problem 1-12
55  print(record1[0]-record2[0])                     # Problem 1-13
56  print(add_four(4, 3))                            # Problem 1-14
```

注意：背面尚有試題

Problem 2 [16%] [2-1 ~ 2-8 each 2%]

The following C program can implement "put(), get(), and printMap_keys()" functions of "**map**" about Python. Please trace the program and fill in the blanks for problems 2-1 ~ 2-8 with correct statements, so that the output is: "NCKU, NCU, NTHU, NCTU, NTU, NTUT, NCKU, NCU, NTHU, NCTU, NTU, =>124, =>-1".

```
01  #include <stdio.h>
02  #include <stdlib.h>
03  #include <string.h>
04  typedef struct Node {
05      char *key;
06      int value;
07      struct Node *next;
08  } Node;
09  Node* make_node(const char* key, int value) {
10      Node* node = malloc(sizeof(Node));
11      node->key = strdup(key);
12      node->value = value;
13      node->next = _____;                          /* Problem 2-1 */
14      return node;
15  }
16  void put(Node **headp, const char* key, int value) {
17      Node *cur = _____;                           /* Problem 2-2 */
18      while (cur != NULL) {
19          if (strcmp(cur->key, key) == 0) {
20              cur->value = value;
21              _____;                               /* Problem 2-3 */
22          }
23          cur = cur->next;
24      }
25      Node *node = make_node(_____);               /* Problem 2-4 */
26      node->next = *headp;
27      *headp = node;
28  }
29  int get(Node *head, const char* _____, int default_value) {   /* Problem 2-5 */
30      Node *cur = head;
31      while (cur != NULL) {
32          if (strcmp(cur->key, key) == 0)
33              return _____;                        /* Problem 2-6 */
34          cur = cur->next;
35      }
36      return _____;                                /* Problem 2-7 */
37  }
38  void printMap_keys(Node* head) {
39      Node* cur = head;
40      while (cur != NULL) {
41          printf("%s, ", _____);                   /* Problem 2-8 */
42          cur = cur->next;
43      }
44  }
45  int main() {
46      Node* map = NULL;
47      char* keys[] = {"NTU", "NCTU", "NTHU", "NCU", "NCKU"};
48      int values[] = {112, 113, 114, 115, 116};
49      for (int i=0; i<5; i++) put(&map, keys[i], values[i]);
50      printMap_keys(map);
51      put(&map, "NTUT", 124);
52      printMap_keys(map);
53      printf("=>%d, ", get(map, "NTUT", -1));
54      printf("=>%d\n", get(map, "Apple", -1));
55      return 0;
56  }
```

Problem 3 [18%] [3-1 ~ 3-5 each 3%, 3-6~3-8 each 1%]
Please trace the following C program and fill in the blanks for problems [3-1~3-8] with correct statements, so that the output is:

```
7 8 6 5 4 1 3 2
Count: 25
1 2 3 4 5 6 7 8
```

```
#include <stdio.h>
void swap(int* a, int* b){
    *a = *a ^ *b;
    *b = _____; // Use XOR to do swap.                    /* Problem 3-1 */
    *a = *a ^ *b;
}
int variant_selection_sort(int* array, int size){
    int count = 0;
    for(int i = 0; i < size; i++){
        for(int j = i; j < size; j++){
            if(array[i] _____){                          /* Problem 3-2 */
                swap(&array[i], &array[_____]);          /* Problem 3-3 */
                count += 1;
            }
        }
    }
    return _____;                                        /* Problem 3-4 */
}
void reverse_array(int* array, int size){
    for(int i = 0; i < size/ ____; i++){                 /* Problem 3-5 */
        swap(&array[i], &array[_____]);                  /* Problem 3-6 */
    }
}
int main(){
    int array[] = {2, 3, 1, 4, 5, 6, 8, _____};          /* Problem 3-7 */
    // use sizeof to get the array size.
    int size = sizeof(_____)/sizeof(int);                /* Problem 3-8 */
    reverse_array(array, size);
    for(int i = 0; i < 8; i++){
        printf("%d ", array[i]);
    }
    int count = variant_selection_sort(array, size);
    printf("\nCount: %d\n", count);
    for(int i = 0; i < 8; i++){
        printf("%d ", array[i]);
    }
}
```

Problem 4 [17%] [4-1 ~ 4-3 each 3%, 4-4 ~ 4-7 each 2%]
This problem is designed to test the understanding of constructors and destructors, object lifetime, member initialization, inheritance, operator overloading, and the difference between prefix and postfix ++. Please trace the following C++ program and fill in the blanks for problems 4-1~4-3 with correct statements, and determine what is printed on the screen for problems 4-4 ~ 4-7. (Ignore newline characters.). For each problem, **choose one from the following options:**
(A) rhs.value (B) value+rhs.value (C) rhs (D) &rhs (E) 9ehb (F) value+*rhs.value (G) value+&rhs.value (H) &rhs.value (I) value (J) c.value (K) 9bhe (L) 10bhe (M) BHE (N) EHB (O) E (P) this (Q) *this (R) c (S) &c (T) value++ (U) 10ehb (V) BHE9 (W) BHEehb (X) 1 (Y) 2 (Z1) 3 (Z2) 4 (Z3) 5 (Z4) 6 (Z5) 7 (Z6) 8 (Z7) 9 (Z8) 10.

```
#include <iostream>
using namespace std;
class CounterBase {
public:
    CounterBase()   { cout << "B"; }
    ~CounterBase() { cout << "b"; }
};
class CounterHelper {
public:
    CounterHelper()   { cout << "H"; }
    ~CounterHelper() { cout << "h"; }
};
class Counter : public CounterBase {
private:
    CounterHelper helper;
    int value;
public:
    Counter(int v = 0) : CounterBase(), helper(), value(v) { cout << "E";}
    ~Counter() { cout << "e"; }
    Counter operator+(const Counter &rhs) const
    { return Counter(_____); }                           /* Problem 4-1 */
    Counter& operator++() {
        ++value;
        return *this;
    }
    Counter operator++(int) {
        Counter temp = _____;                            /* Problem 4-2 */
        ++value;
        return temp;
    }
    friend ostream& operator<<(ostream &os, const Counter &c) {
        os<<_____;                                       /* Problem 4-3 */
        return os;
    }
};
int main() {
    Counter a(5), b(3), c(0);
    c = a + b;
    cout << ++c << endl;          /* Problem 4-4 */
    cout << c++ << endl;          /* Problem 4-5 */
    {   Counter temp(1); }        /* Problem 4-6 */
    cout << c << endl;            /* Problem 4-7 */
    return 0;
}
```

注意：背面尚有試題

Problem 5 [16%] [5-1 ~ 5-8 each 2%]

The following program demonstrates shallow copy and deep copy.

- The **Message** class uses the **default copy constructor** and the **default copy assignment operator**, so it performs **shallow copy**. When a **Message** object is copied or assigned, only the pointer is copied. Therefore, two objects may share the **same content memory**, and modifying one object may also modify the other.
- The **Note** class correctly implements **deep copy**, so every copied object allocates its own memory. Modifying one **Note** object does **not** change the others.

Please trace the execution of the program and determine what each "print()" statement outputs.
For problems (5-1 ~ 5-8), **choose one from the following options:**
(A) 4:3:Hi (B) 4:3:Bye (C) 4:4:Hi (D) 5:6:Hi Bye (E) 6:6:Hi Bye (F) 5:6:Good Bye (G) 5:7:Hi Bye
(H) 5:7:Hi ByeB (I) 8:6:Hi Bye (J) 12:6:Hi Bye (K) 5:10:Good Night (L) 5:10:Good Bye
(M) 10:10:Good Night (N) 12:10:Good Night (O) 12:12:Good Night (P) 12:10:Good NightA
(Q) 12:10:Hi NightA (R) 5:7:Hi ByeA (S) 5:6:Good ByeB (T) 4:5:Hi (U) 6:7:Hi Bye (V) 8:4:Hi B.

```
01  #include <iostream>
02  #include <cstring>
03  using namespace std;
04  class Message {                          //default copy constructor(shallow copy)
05  public:                                  //default copy Assignment Operator (shallow copy)
06      char* content;
07      int capacity;
08      Message(const char* s) {
09          capacity = 5;
10          content = new char[capacity];
11          strcpy(content, s);
12      }
13      void append(const char* extra) { strcat(content, extra); }
14      const void print() {
15          cout<<capacity<<":"<< strlen(content)<<":"<<content<<endl;
16      }
17  };
18  class Note {
19  private:
20      char* content;
21      int capacity;
22      void extendCapacity(int newLen) {                    // Ensures sufficient capacity
23          capacity = capacity*2;
24          if (capacity < newLen + 1) capacity = newLen + 1;
25      }
26      void extendContent(int newLen) {                     // Reallocates if necessary
27          if (newLen + 1 <= capacity) return;
28          extendCapacity(newLen);
29          char* temp = content;
30          content = new char[capacity];
31          strcpy(content, temp);
32          delete [] temp;
33      }
34  public:
35      Note(const char* text) {
36          capacity = strlen(text) + 1;
37          content = new char[capacity];
38          strcpy(content, text);
39      }
40      Note(const Note &other) {                            // deep copy constructor
41          capacity = other.capacity;
42          content = new char[capacity];
43          strcpy(content, other.content);
44      }
45      Note& operator=(const Note &other) {                 // deep copy assignment operator
46          if (this == &other) return *this;
47          delete [] content;
48          capacity = other.capacity;
49          content = new char[capacity];
50          strcpy(content, other.content);
51          return *this;
52      }
53      void append(const char* extra) {                     // add extra content into Note
54          int newLen = strlen(content) + strlen(extra);
55          extendContent(newLen);
56          strcat(content, extra);
57      }
58      const void print() {
59          cout<<capacity<<":"<< strlen(content)<<":"<<content<<endl;
60      }
61  };
62  void testShallowCopy() {
63      Message m1("Good ");
64      Message m2("Hi ");
65      Message m3(m1);
66      m1 = m2;
67      m2.append("Bye");
68      m3.append("Night");
69      m1.print();                                          /* Problem 5-1 */
70      m2.print();                                          /* Problem 5-2 */
71      m3.print();                                          /* Problem 5-3 */
72      m1.append("A");
73      m2.print();                                          /* Problem 5-4 */
74  }
75  void testDeepCopy() {
76      Note n1("Good ");
77      Note n2("Hi ");
78      Note n3(n1);
79      n1 = n2;
80      n2.append("Bye");
81      n3.append("Night");
82      n1.print();                                          /* Problem 5-5 */
83      n2.print();                                          /* Problem 5-6 */
84      n3.print();                                          /* Problem 5-7 */
85      n1.append("B");
86      n2.print();                                          /* Problem 5-8 */
87  }
88  int main() {
89      testShallowCopy();
90      testDeepCopy();
91      return 0;
92  }
```

試題結束