2020 年全國 C++ 等級考試

試題冊

考生注意

1．題目中所有來自 C++ 標準樣板程式庫（C++ Standard Template Library）的名稱都假設已經包含了對應的標頭檔

2．本次考試不考察執行緒和網路程式設計相關內容，考生答題時可以不考慮因執行緒或網路程式設計導致的問題

3．所有寫在試題冊及草稿紙上的答案無效

4．本場考試方式為閉卷筆試，考試時間為 4 個小時

5．考試開始前考生不可打開試題冊

(一)、選擇題 : 1 ~ 8 題, 每小題 2 分, 共計 16 分. 每個選擇題給出 A、B、C、D、E、F 和 G 七個選項, 每一題保證至少有一個正確選項, 至多有七個正確選項. 多選、少選或錯選均不得分.

1. 本試卷考察的內容是 (　　)

A、C++　　B、F#　　C、C#　　D、Objective-C　　E、Fortran
F、機器語言　　G、PHP

2. 在沒有運算子多載的情況下, 下列運算子中有多個含義的是 (　　)

A、&　　B、<　　C、new　　D、~　　E、sizeof...
F、.　　G、typeid

3. 下列選項中, 不支援巢狀的是 (　　)

A、class / struct　　B、enum　　C、函式　　D、/*　*/
E、不具名名稱空間　　F、指示連結　　G、union

4. 設 i 是一個 int 型別且被 constexpr 標識的變數, 下列宣告中正確的是 (　　)

A、decltype(((i))) i2;　　B、decltype(i) &&i3 {4};　　C、int i4(i);
D、void *p = &i;　　E、static int &i5 {i};　　F、const char &&c = i;
G、enum {i, j, k};

5. 下列關鍵字從 C++ 11 才引入的是 (　　)

A、char32_t　　B、register　　C、export　　D、explicit　　E、auto
F、bitor　　G、char8_t

6. 下列選項中, 可能或一定無法通過編碼器編碼的是 (　　)

A、
```
auto empty_pointer = reinterpret_cast<int *>(nullptr);
```
B、
```
class Foo {
    public:
        Foo();
    private:
        void f() = delete;
}
```
C、
```
auto p = new auto {0};
```
D、
```
void f() noexcept {
    throw;
}
```
E、
```
auto arr[] = {1, 2, 3};
```
F、
```
template <typename T>
T f() {
    return {};
}
```
G、
```
constexpr int a = {};
int main(int argc, char *argv[]) {
    #ifdef a
        ++argc;
    #endif
}
```

7. 下列選項中, 可能會發生未定行為的是 (　　)

A、
```
std::vector<int> vec(0, 1, 2, 3, 4, 5, 7);
```

B、
```
std::vector<int> vec(63, 42, std::allocator<int> {});
for(auto c : vec) {
  vec.insert(vec.crbegin() + c, 42);
}
```

C、
```
template <typename T>
typename std::add_pointer<T>::type alloc() {
  return static_cast<typename std::add_pointer<T>::type>(::operator
       new(sizeof T {}));
}
```

D、
```
delete this;
```

E、
```
template <typename T, typename U>
bool compare(const T &t, const U &) {
  return t < u;
}
```

F、
```
int main() {
  class Base {
       ~Base() {}
  };
  class Derived : Base {
       ~Derived() {}
  };
  Derived d;;
}
```

G、設 p 和 q 是一個 int * 型別的指標, 且 p == q, 有 new (p) int(1); 再令 *q = 0;

8. 下列說法正確的是 (　　)

A、std::vector 與 std::string 多載的陣列注標運算子不可以用於添加新的元素

B、std::map 與 std::set 多載的陣列注標運算子可以用於添加新的元素

C、函式可以回傳陣列型別

D、設變數 arr 是 int [N] 型別的陣列, 其中 N 是一個整型常數表達式. 對陣列 arr 使用 decltype 及 auto 進行推導, 得到的型別結果不相同

E、const_cast 運算子對頂層 const 無效

F、可以使用 std::initializer_list 初始化 std::priority_queue

G、不具名 union 存在成員丟失 (即不具名 union 宣告之後無法再使用其成員)

(二)、填空題 : 9 ~ 14 題, 每小題 3 分, 共計 18 分. 每個填空題至少有一個正確答案, 對於多個答案的填空題, 全填對得 3 分, 漏填得 1 分, 存在錯填不得分.

9. 設有

```
int i = 0;
auto &i2 {i};
```

請使用 decltype 宣告變數 i3, 其型別推斷自 i2, 且 i3 不可以被初始化 :

_____.

10. 設有

```
template <typename ...Ts>
struct meta_f {
   constexpr static auto value = true;
};
template <>
struct meta_f<void ()> {
    constexpr static auto value = false;
};
struct Foo {
    constexpr const void *volatile f(int &, char, ...) const
        volatile && noexcept(noexcept(meta_f<typename
            add_lvalue_reference<void>::type>::value)) {
       return nullptr;
   }
};
```

寫出一指標 p 指向 Foo::f, 不可使用 auto、decltype 以及 template :

_____.

11. 寫出隱含的 inline 函式的情況 :

_____.

12. 寫出函式 f 的宣告, 使得其接受僅接受參數為任意大小的 T 型別陣列的參考, 回傳型別從陣列的第 0 個元素推斷 (不可以使用 template 推導陣列的參考和陣列第 0 個元素的型別) :

_____.

13. 設有程式碼 :

```
#include <iostream>
int main(int argc, char *argv[]) {
     std::string str = "123";
     std::cout << str << std::endl;
     std::cin >> str;
     std::cout << str << std::endl;
}
```

若編碼器對上述程式碼的名稱不作特殊處理, 那麼要使得上述程式碼通過編碼器的編碼, 應該額外寫出的陳述式是 :

_____.

14. 設有函式宣告 :

```
void func(unsigned short *);
void func(unsigned);
void func(long long);
void func(...);
```

函式呼叫 func(NULL) 選中的函式是 :

_____.

(三)、改錯題 : 15 ~ 18 題, 每小題 5 分, 共計 15 分. 每個改錯題的錯誤包括但不限於編碼錯誤、未定行為、錯誤結果和不正確的實作, 指出可能存在錯誤的地方並且在不刪除程式碼的情況下更改程式碼使得你認為其可以正確運作. 初始為 5 分, 漏改一處扣 1 分, 改錯一處扣 2 分, 得分不低於 0 分

15.
```
struct Foo {
  int a(0);
  int override {1};
  static int c {3};
  bool operator==(Foo *) = default;
  Foo(const Foo &) = delete;
  Foo(Foo &&) noexcept = default;
  virtual ~Foo() = default;
};
struct Bar : Foo {
  std::string str;
  Bar(const string str = "0") : str(str) {}
};
int main(int argc, char *argv[]) {
  Bar b1("hello");
  Bar b2 = static_cast<Bar &&>(b1);
}
```

16.
```
template <typename T>
struct add_lvalue_reference {
      using type = T &;
};
```

17.
```
void f(char &) noexcept;
int main(int argc, char *argv[]) {
  decltype(f) fp;
  fp = f;
  char c = 0;
  fp(c);
  auto *p = new int;
  const int *cp = nullptr;
  const_cast<int *>(cp) = p;
  struct Foo {
        int *p;
        struct Bar {
              auto i = 0;
              bool operator(char c)() {
                    if(this->i == c) {
                          return false;
                    }
                    fp(c);
                    return true;
              }
        };
  };
}
```

18. 在某一次競賽中，試題卷中其中一試題為：計算 $\frac{1}{1!}-\frac{1}{3!}+\frac{1}{5!}-\ldots+\frac{(-1)^{n+1}}{(2n-1)!}$。其中，n 的值由用戶給定。某位同學給出的程式碼如下：

```
long long fac(int n, int r = 1) {
    return fac(--n, r * n);
}
long double cal(int n) {
    bool symbol = true;
    long double r = 0.0;
    for(int d = 1; d <= n; d += 2) {
        r += []() mutable {
            if(symbol) {
                return 1 / static_cast<long double>(fac(d));
            }
            return -1 / static_cast<long double>(fac(d));
        }
        symbol = -symbol;
    }
    return r;
}
```

上述程式碼存在錯誤，在不更改函式 fac 與 cal 型別、不刪除任何函式內的變數、不修改函式內已有變數的型別及保持 lambda 表達式的情況下修改上述函式，使得其 cal(n) 可以得到正確的結果。

(四)、論述題 : 19 ~ 24 題, 每小題 2 分, 共計 12 分.

19. 簡述 C++ 標準樣板程式庫中的 I/O 物件不可被複製的原因.
20. 簡述移動建構子或移動指派運算子應儘量標識 noexcept 的原因.
21. 簡述被移動的物件不建議使用的原因.
22. 簡述 std::list 與 std::forward_list 有自己的排序函式的原因.
23. 簡述 C++ 標準樣板程式庫中的演算法通常要求傳入疊代器而非容器的好處.
24. 使用 5 種不同的方法產生可呼叫物件, 其中函式原型為

```
const Foo ::operator==(Bar &, Baz *) noexcept;
```

(五)、程式設計 : 25 ~ 30 題, 共計 39 分. 本題除了小題有額外的要求之外, 要求全程自行設計, 不可使用類似於 C++ 標準樣板程式庫等中的任何名稱.

25. 特性創造題 : ① ~ ③ 題, 每題 2 分. 下列都是未被 C++ 11 引入的特性, 請閱讀這些程式碼, 寫出這些特性並且介紹特性的用處.

①. 設有函式原型 : Mat visual::threshold(Mat, int, int, int);
多數使用者在使用函式時, 呼叫的形式為 :
visual::threshold(matrix, default, default, 188);

②. 設 q 為 int * 型別的指標, 使用者的程式碼 :

```
if(auto p {q}; q) {
      //do something...
}
```

③. 設下列程式碼位於某個類別內, 類別內有成員變數 p 與 f, 它們都可以隱含地轉型為 bool 型別 :

```
auto lambda {[this](auto a, auto b) noexcept -> auto {
      if(this->p) {
            return true;
      }
      return this->f;
}};
```

26. 資料結構實作題 : 6 分. 堆疊 (stack) 是一種滿足先進後出 (FILO) 的資料結構. 每一次入堆疊都是將元素放入堆疊的頂部, 每一次出堆疊都是將堆疊頂部的元素從堆疊中刪除. 要求使用 class 實作一個堆疊, 它至少滿足元素入堆疊 (push), 元素出堆疊 (pop), 堆疊的初始化 (給定一個 size, 使得堆疊的大小為 size, 當 size 未指定, 則預設大小為 64), 堆疊的清空 (clear), 判定堆疊是否為空 (empty), 取堆疊頂部的元素 (top). 對於出現的邊界情況, 考慮擲出例外情況.

27. 物件導向設計題 : 6 分. 生物 (creature) 是地球 (Earth) 上迄今為止最為神奇的一類, 也是大自然 (nature) 的饋贈 (gift), 萬物生之於大自然, 止之於大自然. 生物有生命(lifetime)、呼吸 (breathe)、身長 (height) 和體重 (weight) 等生物特徵(biometric). 生物不斷進化 (evolution), 演化出了靈長類 (primate). 靈長類擁有眼(eyes)、鼻 (nose)、嘴 (mouse)、耳 (ears)、手(hand)、腳 (foot), 除此之外, 靈長類還進化出了爬行 (crawl) 等能力. 靈長類和我們息息相關, 靈長類不斷進化有了猴(monkey) 和其餘的物種 (other creature). 猴子除了擁有靈長類的基本特徵之外, 還有了尾巴 (tail) 和體毛 (hair) 等特徵, 還擁有了跳躍 (jump) 等的能力 (ability). 猴子不斷進化, 就有了古猿 (acient ape), 它的尾巴退化 (degeneration) 了. 古猿的進化有分化, 一部分古猿進化成了猿 (ape), 還有一部分進化成了原始人 (primitive), 它們都擁有直立行走 (walk) 和跑 (run) 等的能力. 猿和原始人共同進化, 就有了智慧 (wisdom) 人(human). 人類擁有智慧之後, 就學會了自我獨立思考 (think), 語言 (language) 表達(speak) 等的能力, 創造 (create) 了很多東西 (things), 開始給自己取名字 (name), 創造出了社會 (society), ..., 直到現代世界 (modern world). 請根據下列要求, 設計類別 :

①可以使用的 C++ 標準樣板程式庫物件 : std::string, std::cin, std::cout, std::endl

②每一個動作可以使用一個輸出陳述式替換, 不可以出現僅僅宣告而不實作的函式

③生物屬性不應向外公開

28. 遞迴設計題 : 6 分. 河內塔 (Hanoi Tower) 問題 : 傳說越南河內某間寺院有三根銀棒, 上串 N (N 由使用者輸入) 個金盤. 寺院裡的僧侶依照一個古老的預言, 以下面列出的規則移動 64 個盤子. 預言說當這些盤子移動完畢，世界就會滅亡. 設初始這些金盤按順序被放置在第一根銀棒 X 上, 設第二根銀棒是 Y, 設第三根銀棒為 Z. 每一次只能移動一個金盤, 要求金盤全程由小到大排列 (即不能出現大的金盤放置在小的金盤上), 所有金盤最終需要按順序放置在銀棒 Z 上. 設計程式, 輸出每一次移動的過程 (例如從 X 移動到 Y), 統計總共移動的次數, 並且輸出最終銀棒 Z 上的金盤情況. 有如下要求 :

①可以使用 C++ 標準樣板程式庫的物件 : std::vector, std::string, std::deque, std::cin, std::cout, std::endl

②不可使用 std::vector 及 std::string 的陣列注標運算子

③要求使用過程導向的程式設計方式

④要求使用遞迴的方式實作

29. 過程導向設計題 : 6 分, ① ~ ④ 題.

①(2 分) 設有函式 :

```
template <typename ...Args>
void f(Args &&...);
```

設包 Args 中的型別可以向 int 轉型. 設計程式, 寫出函式 f 對包 Args &&... 中的引數進行排序後, 輸出這些引數的整型資料. 可以使用的 C++ 標準樣板程式庫物件 : std::cout, std::endl, std::forward

②(1 分) 設函數 (function) $f(x) = a\,x^n$, 它的導數 (derivative) 表達式為 $f(x) = n \cdot a\,x^{n-1}$. 設計函式計算任意給出的 $f(x)$ 的導數, 若給定 x, 則需要回傳計算結果. 可以使用的 C++ 標準樣板程式庫物件 : std::pow

③(1 分) 設函數 (function) $f(x) = a\,x^n$, 它的原函數 (primitive) 表達式為 $F(x) = \frac{a}{n}x^{n+1}$, $f(x)$ 在區間 $[a, b]$ 上的定積分 (definite integral) 的表達式為 $\int_b^a f(x)dx = F(b) - F(a)$, 沒有給出區間的積分 (integral) 稱為不定積分, 它就是結果就是原函數加上一個任意常數 (本題中省略這個常數). 請設計函式計算任意給定的 $f(x)$ 函數的不定積分. 若給出區間, 則需要回傳定積分的計算結果. 可以使用的 C++ 標準樣板程式庫物件 : std::pow

④(2 分) 廣義表是 Lisp 中內建的資料結構, C++ 中廣義表非內建, 但是可以通過自行實作的方式獲得廣義表. 廣義表中的元素既可以是單個的, 又可以仍然是一個廣義表. 設使用 C++ 實作的廣義表 (generalized_list) 有如下操作 : 取頭部元素 (top), 取尾部元素 (back), 取第 n 個元素 (operator[]), 移除第 n 個元素 (erase(n)), 判定某個元素是否為廣義表 (is_glist(n)), 計算廣義表中的元素數量 (size), 判定廣義表是否為空 (empty), 如若某個元素不是廣義表而是單個元素, 那麼它仍然以廣義表存儲, 只不過他可以向對應的型別進行隱含型別轉化, 單個元素也可以向廣義表進行隱含型別轉換. 設計一函式展開廣義表 :

```
std::vector<int> unfold(const generalized_list<int> &) noexcept;
```

例如某個廣義表為 (1, (1, 2, 3), (1, (2, 3), (3, (4, 5, ()))))), 展開後回傳的 std::vector 中應保存 {1, 1, 2, 3, 1, 2, 3, 3, 4, 5}. 可以使用的 C++ 標準樣板程式庫中的物件為 std::vector, std::stack, 不可以使用遞迴的方式實作

30. 記憶體控制題 : 6 分. 現在有一聯合開發的項目, 項目使用的程式庫使用的是舊式 C++ 標準, 而這個項目要求使用 C++ 標準樣板程式庫的智慧指標 std::unique_ptr. 在項目使用的程式庫中, 某一個函式要求給定一個型別為 T ** 的引數, 但是 std::unique_ptr 本身並不提供一個成員函式來回傳一個指標的參考. 某個同學想到使用 reinterpret_cast 來實作 :

```
#include <memory>
void f(T **);
int main(int argc, char *argv[]) {
      std::unique_ptr<T> p = new T();
      f(reinterpret_cast<T **>(&p));
}
```

在某些情況下, 這種方法確實可行, 但是這樣容易產生未定行為甚至嚴重錯誤, 程式碼的可攜性與相容性也得不到保障. 現在需要設計一個類別, 對上述需求進行包裝, 提供易用性、安全性和可攜性. 有如下要求 :

①可使用的 C++ 標準樣板程式庫物件 : std::unique_ptr, std::default_delete, std::exchange. 其中, std::exchange 可能的實作為

```
template<class T, class U = T>
T exchange(T& obj, U&& new_value)
{
    T old_value = std::move(obj);
    obj = std::forward<U>(new_value);
    return old_value;
}
```

②不可以重新設計 std::unique_ptr 或著更改 std::unique_ptr, 只能設計一個輔助類型的類別