


<b>Giảng viên ra đề:</b> (Chữ ký và Họ tên)	(Ngày ra đề)	<b>Người phê duyệt:</b> (Chữ ký và họ tên)	(Ngày duyệt đề)
		39/45	

  TRƯỜNG ĐH BÁCH KHOA - ĐHQG-HCM <u>KHOA KH &amp; KT MÁY TÍNH</u>	<b>KT GIỮA KỲ</b>		Học kỳ / Năm học	2	2023-2024
			Ngày thi	12-03-2024	
	Môn học	Nguyên lý ngôn ngữ lập trình			
	Mã môn học	CO3005			
	Thời lượng	60 phút	Mã đề	2320	
<b>Ghi chú:</b> <ul style="list-style-type: none"><li>- Sinh viên làm bài trên phiếu trả lời trắc nghiệm. KHÔNG được phép dùng tài liệu.</li><li>- Các câu hỏi chỉ có 1 đáp án đúng hoặc không có đáp án đúng.</li><li>- Nếu không có đáp án đúng, sinh viên chọn đáp án E.</li><li>- Sinh viên nộp đề cùng với phiếu trả lời trắc nghiệm sau khi kiểm tra.</li><li>- Tất cả câu hỏi có mã [A1-1] và [A1-2] cũng sẽ được dùng để tính toán Bài tập lớn 1.</li><li>- Tất cả câu hỏi có mã [A2] cũng sẽ được dùng để tính toán Bài tập lớn 2.</li></ul>					

**Câu 1.** [L.O.2.1] Giả sử một chương trình có n dòng lệnh nhưng do có các phát biểu rẽ nhánh và lặp nên sẽ có m dòng lệnh được thực thi. Giả sử mỗi dòng lệnh đều cần t1 giây để dịch và đều cần t2 giây để thực thi. Hãy chọn công thức tính thời gian dịch và thực thi lần đầu tiên chương trình trên khi dùng trình biên dịch và trình thông dịch?

biên dịch:  $n*t1 + m*t2$

- (A) Trình thông dịch:  $m*t1 + m*t2$   
 (C) Trình thông dịch:  $n*(t1+t2)$

- (B) Trình biên dịch:  $m*t1 + m*t2$  thông dịch:  $(t1 + t2)*m$   
 (D) Trình biên dịch:  $n*(t1+t2)$

**Câu 2.** [L.O.2.1] Với một khai báo macro trên C++ như:

```
#define MAX 50
```

Chương trình nào sẽ thay các tên macro MAX xuất hiện trong chương trình bởi giá trị (50) của nó?

- (A) ~~Biên dịch (Compiler)~~  
 (C) Trình hợp ngữ (Assembler)

- (B) Tiền xử lý (Preprocessor)  
 (D) Trình liên kết (Link editor)

Trong C++, tiền xử lý (Preprocessor) là chương trình chạy đầu tiên trước khi biên dịch. Nó có nhiệm vụ xử lý các chỉ thị tiền xử lý, bắt đầu bằng dấu #, ví dụ như #define, #include, #ifdef, v.v.

**Câu 3.** [L.O.2.1] Trình biên dịch đúng lúc (just-in-time compiler) là một thành phần của:

- (A) Trình biên dịch (B) Trình tiền xử lý (C) Trình hợp ngữ (D) Trình thông dịch

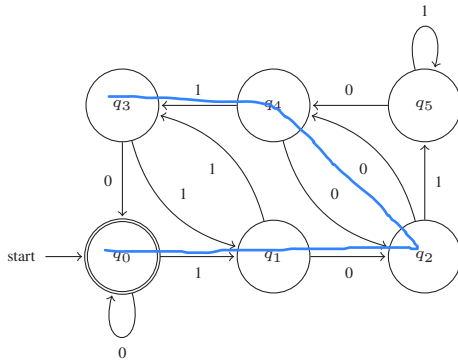
Trình biên dịch đúng lúc (Just-In-Time Compiler - JIT) là một thành phần quan trọng của trình thông dịch (Interpreter) hiện đại.

Trình thông dịch truyền thống: Dịch và thực thi từng dòng lệnh mỗi khi chương trình chạy, dẫn đến hiệu suất chậm. JIT Compiler: Khắc phục nhược điểm này bằng cách biên dịch "đúng lúc", tức là trong quá trình chương trình đang chạy. Nó sẽ nhận diện các đoạn code được thực thi nhiều lần (hotspot) và biên dịch chúng sang mã máy để tăng tốc độ thực thi.

Tại sao các đáp án khác không đúng:

- A. Trình biên dịch (Compiler): Trình biên dịch truyền thống biên dịch toàn bộ chương trình trước khi chạy. JIT compiler hoạt động khác, nó biên dịch trong quá trình chạy.  
 B. Trình tiền xử lý (Preprocessor): Trình tiền xử lý xử lý các chỉ thị tiền xử lý (ví dụ: macro, include) trước khi biên dịch. Nó không liên quan đến việc biên dịch "đúng lúc" trong quá trình chạy.  
 C. Trình hợp ngữ (Assembler): Trình hợp ngữ dịch mã hợp ngữ sang mã máy. Nó cũng là một giai đoạn biên dịch, nhưng không phải là JIT compiler.

**Câu 4.** [L.O.1.1] Cho một automata biểu diễn chuỗi nhị phân như sau:



Một số nhị phân chia hết cho 2 nếu chữ số cuối cùng của nó là 0

Cho các phát biểu sau:

- (a) Giá trị thập phân của chuỗi nhị phân khác rỗng sinh ra bởi automata trên đều chia hết cho 2. ✓  
 (b) Automata trên có thể sinh ra được tất cả các chuỗi nhị phân có dạng  $101^*0$ . ✗  
 (c) Tất cả các chuỗi khác rỗng sinh ra bởi automata trên có độ dài bé hơn 4 đều có giá trị thập phân của nó chia hết cho 6. ✓  
 (d) Mọi chuỗi sinh ra bởi automata trên bằng cách đi qua tổ hợp các trạng thái  $q_0, q_1, q_2, q_4, q_3$  đều phải có số lẻ ký tự 0. ✗  
 (e) Có đúng 1 chuỗi có độ dài bé hơn 5 sinh ra bởi automata trên mà giá trị thập phân của nó không chia hết cho 3. ✗

0 chuỗi nào?

câu 5: Tương đương  $(x|y)^*(a|ab)^*$  độ dài < 3  
 $x, y, a, xx, yy, aa, ab, xa, ya$ , rỗng,

Số phát biểu đúng là

(A) 2

(B) 1

(C) 4

$xy, yx$

(D) 3

**Câu 5.** [A1-1] Số chuỗi có độ dài nhỏ hơn 4 được sinh ra bởi biểu thức chính quy  $(x|y)^*y(a|ab)^*$  là:

(A) 7

(B) 12

(C) 10

(D) 11

$y, xy, yy, ya, yab, xya, yya, xxy, yyy, yaa, yxy, xyy$

**Câu 6.** [A1-1] Cho M là ngôn ngữ chứa các chuỗi không rỗng của các ký tự chữ thường (a-z), trong đó nếu một chuỗi bắt đầu bằng ký tự 'u' thì không được kết thúc bằng ký tự 'u'. Biểu thức chính quy mô tả M là gì?

(A)  $u^*[a-tv-z]^*| [a-tv-z]^*u^*$

(B)  $u^+[a-tv-z]^*u^*$

(C)  $[a-tv-z][a-z]^*|u[a-z]^*[a-tv-z]$

(D)  $[a-tv-z][a-z]^*|u[a-z]^+[a-tv-z]$

**Câu 7.** [A1-1] Tên tài khoản trên mạng xã hội Instagram được quy định như sau:

- Có ít nhất 1 ký tự.
- Bao gồm các ký tự thường (a-z), ký tự số (0-9), dấu gạch dưới (\_) và các dấu kết thúc (?, !, .)
- Không được bắt đầu hoặc kết thúc bằng các dấu kết thúc
- Không có hai dấu kết thúc liên tiếp nhau.

Biết rằng, L, D, U, P là tên các fragment đã được định nghĩa tương ứng cho ký tự thường, ký tự số, dấu gạch dưới và dấu kết thúc trong ANTLR4. Biểu thức chính quy mô tả tên tài khoản Instagram là gì?

(A)  $(L|D|U) (P (L|D|U))^*$

(B)  $((L|D|U) (P (L|D|U))^+)^*$

(C)  $(L|D|U)^+ (P (L|D|U))^+^*$

(D)  $(L|D|U)^+ (P (L|D|U))^*^*$

**Câu 8.** [L.O.1.1] Cho một số công dụng sau đây:

- (a) Kiểm tra các ràng buộc như biến phải được khai báo trước khi sử dụng
- (b) Xác định cấu trúc của chuỗi tokens có phù hợp không
- (c) Tách chuỗi nhập thành các chuỗi con ứng với các tokens
- (d) Loại bỏ các chuỗi con ứng với khoảng trắng (như dấu blank, tab, chú thích, ...)
- (e) Gắn thông tin vị trí (hàng, cột) vào mỗi token

Số vai trò của bộ phân tích từ vựng trong các công dụng trên là

- (A) 1                      (B) 2                      (C) 3                      (D) 0

**Câu 9.** [A1-1] Chọn biểu thức chính qui tương đương với biểu thức chính qui sau:  $a^*(ba^*b)+a^*$

- (A)  ~~$(a^*bba^*)^*a^*bba^*$~~  *ababa*      (B)  $a^*ba^*(bba^*)^*ba^*$       (C)  ~~$a^*(bb)+a^*$~~       (D)  ~~$a^*(ba^*ba^*)^+$~~  *abaabbaaba*  
*babababa*

**Câu 10.** [A1-1] Gọi L là ngôn ngữ gồm các chuỗi khác rỗng trên {a,b} mà không có 3 ký tự b liên tiếp. Hãy chọn biểu thức chính qui mô tả ngôn ngữ L?

- (A)  ~~$(a \mid bb^*a)+b^*b^? \mid bb^?$~~       (B)  $a+(b^?b^?(ab)^?)^* \mid bb^?$  *abba*  
(C)  ~~$a^*(bb^?a^+)+b^?b^? \mid bb^?$~~  *a*      (D)  ~~$a^*(b^?a^*b^?(ab)^?a^*)+b^?b^?$~~  *aabaa*  
*rỗng*

Áp dụng mô tả sau cho các câu 11–12:

Cho ngôn ngữ X có các mô tả từ vựng trên ANTLR như sau:

```
ID: [a-z]+;  
SEP: [[\] () , ] ;  
NUM: [0-9]+ ('.' [0-9]+)? ;  
OP: [+ \- * > = < %] ;  
WS: [ \r \n \t ]+ -> skip;
```

Cho đoạn mã sau được viết trên ngôn ngữ X:

```
res += (lst[0] * 2) + func(x, y) - (lst[-1] if lst[1] >= -1.2 else lst[2]) % 5
```

**Câu 11.** [A1-1]

Số token được phân tích từ vựng trả về khi phân tích từ vựng cho chuỗi trên là

- (A) 38                      (B) 40                      (C) 42                      (D) Một giá trị khác hoặc lỗi

**Câu 12.** [A1-1]

Chuỗi lexeme của token thứ 25 là:

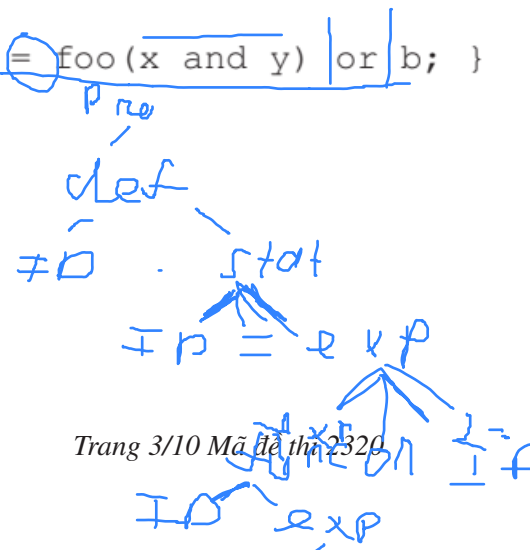
- (A) -1                      (B) ]                      (C) if                      (D) lst

Áp dụng mô tả sau cho các câu 13–15:

Cho ngôn ngữ X được mô tả trong ANTLR4 như sau:

```
1 grammar X;  
2 program: stat EOF | defi EOF; 1 stat hoặc 1 defi  
3 stat: ID '=' expr ';' | expr ';';  
4 defi: ID '(' ID '(' ID ')' stat* ')' ;  
5 expr: ID | INT | func | 'not' expr | expr 'and' expr | expr 'or' expr;  
6 func: ID '(' expr '(' expr ')' ;  
7 INT: [0-9]+;  
8 ID: [a-zA-Z_] [a-zA-Z_0-9]* ;  
9 WS: [ \t \n \r \f ]+ -> skip;
```

$f(x, y) \{ a = \text{foo}(x \text{ and } y) \text{ or } b; \}$



Câu 13. [A1-2] Cho các đoạn mã sau:

- (a) `x and y; f(1);` ✗  
 (b) `f(x, y) { a = 3 + foo; x and y; }` ✗  
 (c) `b = 1 and foo(1, 2);` ✓  
 (d) `f(x, y) { a = 3 or foo(x and y); }` ✓  
 (e) `f(x, y) { a = 3 or foo(); }` ✗

Số đoạn mã thoả ngôn ngữ X là

- (A) 1 (B) 3 (C) 4 (D) 2

Câu 14. [A1-2] Với chuỗi nhập `x and y; f(1);`, bộ phân tích ngữ pháp theo mô tả trên sẽ tạo ra kết quả nào sau đây?

- (A) Error on line 1 col 9: f (B) Error on line 1 col 11: 1  
 (C) Error on line 1 col 7: ; (D) successful

Câu 15. [A1-2] Với chuỗi nhập `f(x, y) { a = foo(x and y) or b; }`, cây phân tích cú pháp sinh ra bởi văn phạm trên có chiều cao là

- (A) 10 (B) 9 (C) 8 (D) 7

Câu 16. [A1-2] Cho một ngôn ngữ bao gồm các khai báo biến được mô tả trong ANTLR4 như sau:

```
1 program: decl EOF;
2 decl: INT ID (CM ID)* SM;
3 ID: [a-z]+; INT: 'int'; CM: ','; SM: ' ';
4 WS: [ \n\r\f\t] -> skip;
```

Nếu một câu lệnh khai báo chứa các định danh trùng lặp (ví dụ: `int x, y, x;`), bộ phân tích cú pháp sẽ xử lý như thế nào?

- (A) Nó sẽ phân tích thành công, xem xét các định danh trùng lặp là các biến riêng biệt.  
 (B) Nó sẽ phát sinh lỗi cú pháp vì các định danh phải là duy nhất trong một câu lệnh khai báo.  
 (C) Nó sẽ phát sinh lỗi cú pháp vì không có luật sinh phù hợp.  
 (D) Nó sẽ phân tích thành công, lỗi ngữ nghĩa trong quá trình thực thi.

do ID được lexer bắt trước, nên "int" cũng được nhận dạng là ID.

Câu 17. [A1-2] Câu nào sau đây đúng về ngôn ngữ sinh ra bởi văn phạm phi ngữ cảnh sau?

$s \rightarrow aCb, a \rightarrow Aa|e, b \rightarrow Bb|B$

- (A) Ngôn ngữ này gồm tất cả các chuỗi trên ngôn ngữ với tập ký tự  $\{A, B, C\}$   
 (B) Ngôn ngữ này gồm tất cả các chuỗi dạng  $A^mCB^n$ , trong đó  $m \geq 0, n \geq 1$   
 (C) Ngôn ngữ này gồm chỉ chuỗi rỗng  
 (D) Ngôn ngữ này gồm tất cả các chuỗi dạng  $A^nCB^{n+1}$ , trong đó  $n$  là số nguyên không âm.

Câu 18. [A1-2] Một phát biểu có thể là phát biểu ghép hoặc phát biểu gán. Một phát biểu ghép được viết bởi

- Token LP rồi đến RP hoặc
- Một danh sách không rỗng các phát biểu được đặt giữa LP và RP.

danh sách có thể rỗng

Gọi `stmt` và `assign` lần lượt mô tả cho một phát biểu và phát biểu gán, các ví dụ sau là các phát biểu hợp lệ: `assign, LP RP, LP assign LP RP RP ...` Hãy chọn các vế phải phù hợp cho `stmt`?

- (A) `LP RP | LP (assign | LP RP)+ RP | assign` (B) `LP stmt* RP | assign`  
 (C) `LP RP | LP stmt RP | assign` (D) `LP RP | LP stmt stmt RP | assign`

**Câu 19.** [A1-2] Cho `mlist` mô tả một danh sách (có thể rỗng) các biểu thức (`expr`) cách nhau bằng dấu CM. Văn phạm của `mlist` được viết như sau:

(a) `mlist: expr CM mlist | expr | ;` CM ở cuối

(b) `mlist: (expr (CM expr)*)?;`

(c) `mlist: nnmlist | ;`  
`nnmlist: expr CM nnmlist | expr ;`

(d) `mlist: expr etail | ;`  
`etail: CM expr etail | ;`

Trong các cách viết văn phạm trên, có bao nhiêu cách viết ĐÚNG?

(A) 4

(B) 2

(C) 1

(D) 3

**Câu 20.** [A1-2] Cho các mô tả văn phạm sau viết trên ANTLR

`decl: M decl_tail | ;`  
`decl_tail: N decl | P M N;`

có thể rỗng

Hãy chọn về phải phù hợp cho luật sinh `decl` để tương đương các mô tả trên?

(A) ~~(M N)\* M P M N~~

(B) (M N M (P M)\* N)?  
MN

(C) (M (N M)\* P M N)?  
MN

(D) (M N)\* (M P M N)?

Áp dụng mô tả sau cho các câu 21–22:

Cho một đoạn mã trong ngôn ngữ Python như sau:

```
1 class Shape:
2     def area(self): return 0
3 class Square(Shape):
4     def __init__(self, s): self.s = s
5     def area(self): return self.s ** 2
6 class Circle(Shape):
7     def __init__(self, r): self.r = r
```

```
8 def area(self): return 3.14 * self.r ** 2
9 def calculate_total_area(shapes):
10     total_area = 0
11     for shape in shapes: total_area += shape.area()
12     return total_area
13 square, circle = Square(5), Circle(3)
14 print(calculate_total_area([square, circle]))
```

**Câu 21.** [L.O.2.1] Trong ngữ cảnh của đoạn mã trên, tính chất nào của lập trình hướng đối tượng được thể hiện mạnh mẽ nhất:

(A) Single inheritance

(B) Multiple inheritance

(C) Parametric Polymorphism

(D) Subtyping polymorphism

template...

**Câu 22.** [L.O.2.1] Kết quả in ra màn hình của đoạn mã trên là

(A) 53.26

(B) 43.84

(C) 28.26

(D) 34.0

Áp dụng mô tả sau cho các câu 23–24:

Cho một đoạn mã trong ngôn ngữ Python như sau:

```
1 class A:
2     def show(self): return "A"
3 class B(A):
4     def show(self): return "B"
```

```
6 class C(A):
7     def show(self): return self.__class__.__name__
8 class D(C, B): pass DCBA
9 class E(B, C): pass EBCA -> ECBA
10 class F(E, B): pass FEBCAo -> FECBA
```

**Câu 23.** [L.O.2.1] Thứ tự phân giải phương thức (MRO) của F là

(A) F, D, E, B, C, object

(B) F, E, B, C, A, object

(C) F, E, D, B, A, object

(D) F, E, C, B, D, object

MRO(A) = [A] + MRO[OBJECT] + OBJECT = [A, OBJECT]

MRO(B) = [B] + MRO[A] + [A] = [B] + [A, OBJECT] + [A] = [B, A, OBJECT]

MRO(C) = [C] + MRO[A] + [A] = [C] + [A, OBJECT] + [A] = [C, A, OBJECT]

MRO(D) = [D] + MRO[C] + MRO[B] + [B, C] = [D] + [C, A, OBJECT] + [B, A, OBJECT] = [D, C, B, A, OBJECT]

MRO(E) = [E] + MRO[B] + MRO[C] + [B, C] = [E] + [B, A, OBJECT] + [C, A, OBJECT] + [B, C] = [E, B, C, A, OBJECT]

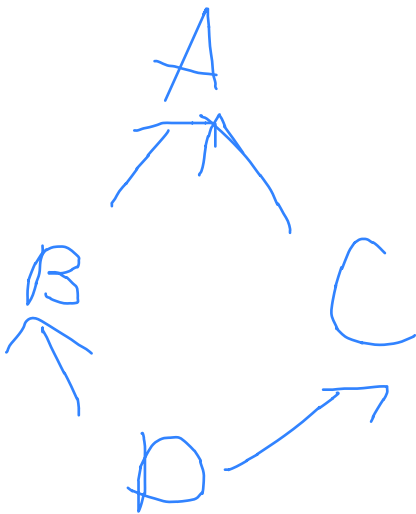
MRO(F) = [F] + MRO[E] + MRO[B] + [E, B] = [F] + [E, B, C, A, OBJECT] + [B, A, OBJECT] + [E, B] = [F, E, B, C, A, OBJECT]

```
>>> 0 = object
>>> class F(0): pass
>>> class E(0): pass
>>> class D(0): pass
>>> class C(D,F): pass
>>> class B(E,D): pass
>>> class A(B,C): pass
```

MRO[F] = [F] + MRO[OBJECT] + OBJECT = [F] + OBJECT + OBJECT = [F, OBJECT]  
MRO[E] = [E] + MRO[OBJECT] + OBJECT = ... = [E, OBJECT]  
MRO[D] = [D, OBJECT]

MRO[C] = [C] + MRO[D] + MRO[F] + [D, F] = [C] + [D, OBJECT] + [F, OBJECT] + [D, F] = [C, D, F, OBJECT]  
MRO[B] = [B] + MRO[E] + MRO[D] + [E, D] = [B] + [E, OBJECT] + [D, OBJECT] + [E, D] = [B, E, D, OBJECT]  
MRO[A] = [A] + MRO[B] + MRO[C] + [B, C] = [A] + [B, E, D, OBJECT] + [C, D, F, OBJECT] + [B, C]  
= [A, B, E, C, D, F]

B là đầu không là đuôi  
-> [E, D, OBJECT] + [C, D, F, OBJECT] + [C]  
E là đầu, không là đuôi  
-> [D, OBJECT] + [C, D, F, OBJECT] + [C]  
C là đầu, không là đuôi  
-> [D, OBJECT], [D, F, OBJECT] + []  
...



D(B,C)

MRO[A] = [A] + MRO[OBJECT] + OBJECT = [A, OBJECT]  
MRO[B] = [B] + MRO[A] = [B] + [A, OBJECT] = [B, A, OBJECT]  
MRO[C] = [C] + MRO[A] = [C] + [A, OBJECT] = [C, A, OBJECT]  
MRO[D] = [D] + MRO[B] + MRO[C] + [B, C]  
= [D] + [B, A, OBJECT] + [C, A, OBJECT] + [B, C]  
= [D, B, C, A, OBJECT]

**Câu 24.** [L.O.2.1] Để kết quả trả về của `F().show()` là "F" thì cần thực hiện gì sau đây?

- Ⓐ Đoạn mã trên đã cho kết quả `F().show()` là "F"
- Ⓑ Thêm lớp cơ sở C vào cuối danh sách hiện tại của lớp F
- Ⓒ Xoá đi lớp cơ sở B trong lớp F
- Ⓓ Thay đổi thứ tự lớp cơ sở của E thành C, B

**Câu 25.** [L.O.2.1] Giả sử mã nguồn của file `lib.py` như sau:

```
class A:
    def f(self):
        print('f in A')
class B(A):
    def g(self):
        super().f()
```

Để dùng được phương thức `g()` của lớp B nhưng thay đổi chức năng của `f()`, một lớp C với phương thức `f` mới và một lớp D mới được viết trên Python như sau:

```
from lib import *
class C():
    def f(self):
        print('f in C')
class D():pass
D().g()
```

Hãy cho biết cần phải khai báo lớp cha của C và D như thế nào để dòng lệnh `D().g()` sẽ có kết quả là 'f in C'?

- Ⓐ `class C(B):` và `class D(A,C):`
- Ⓑ `class C(B):` và `class D(C,A):`
- Ⓒ `class C(A):` và `class D(C,B):`
- Ⓓ `class C(A):` và `class D(B,C):`

**Câu 26.** [L.O.2.1] Cho các lớp dữ liệu `Expr` (mô tả biểu thức tổng quát), `BinExpr` (mô tả biểu thức nhị phân), `UnExpr` (mô tả biểu thức đơn phân), `IntLit` (mô tả hằng nguyên). Giả sử đã có các lớp `Visitor` thực hiện các tác vụ trên các lớp dữ liệu này: `Eval` (tính toán và trả về kết quả của biểu thức), `Prefix` (trả về chuỗi dạng tiền tố của biểu thức). Để chỉ thay đổi chức năng của phương thức `visitUnExpr` trong lớp `Prefix` (các phương thức khác không đổi), theo đó, phương thức này cần trả về chuỗi ứng với giá trị của biểu thức thay vì trả về chuỗi dạng tiền tố của biểu thức, một lớp `PrefixModified` là lớp con của `Prefix` được tạo ra với duy nhất một phương thức `visitUnExpr` như sau:

```
class PrefixModified(Prefix):
    def visitUnExp(self, ctx:UnExpr):
        return _____
```

**Eval và Prefix là visitor -> `Eval().visit(ctx)`**

Ví dụ: `BinExpr(IntLit(3), "+", UnExpr("-", Binary(IntLit(2), "-", IntLit(4))))`.accept(`PrefixModified()`) trả về chuỗi `"+ 3 2"` trong đó 2 là giá trị của `UnExpr("-", Binary(IntLit(2), "-", IntLit(4)))`.

Hãy chọn mã phù hợp để điền vào chỗ trống sau lệnh `return` trong thân của phương thức này để thực hiện được yêu cầu trên?

- Ⓐ `str(Eval().visit(ctx))`
- Ⓑ `str(self.visit(ctx))`
- Ⓒ `str(ctx.visit(Eval()))`
- Ⓓ `str(ctx.accept(self))`

**Câu 27.** [L.O.2.1] Cho các khai báo sau được viết trên một ngôn ngữ lập trình hướng đối tượng dùng kiểm tra kiểu tĩnh như Scala:

```
class A { def foo() = print("a") }
class B extends A { } // B is a subclass of A
class C extends A { override def foo() = print("c") } // C is a subclass of A
class D extends B { override def foo() = print("d") } // D is a subclass of B
```



Biết rằng, biến `b` được khai báo kiểu `B` và đang tham chiếu đến một đối tượng nào đó. Cho một số nhận định về kết quả được in ra khi gọi `b.foo()`

- (a) `c` (nếu `b` đang tham chiếu đến một đối tượng `C`)      (c) `a` (nếu `b` đang tham chiếu đến một đối tượng `A`)  
**(b) `d` (nếu `b` đang tham chiếu đến một đối tượng `D`)**      **(d) `a` (nếu `b` đang tham chiếu đến một đối tượng `B`)**

Các nhận định đúng là

- ☐ (A) (a),(b),(c) và (d)      **☒ (B) (b) và (d)**      ☐ (C) (c) và (d)      ☐ (D) (b) và (c)

**Câu 28.** [L.O.2.1] Khái niệm "decorator" trong Python có thể được sử dụng để làm gì?

- ☒ (A) Thay đổi hoặc mở rộng hành vi của một hàm**      ☐ (B) Thêm chức năng mới vào một hàm  
☐ (C) Định nghĩa một hàm mới từ một hàm thư viện khác      ☐ (D) Xóa chức năng được chỉ định cụ thể từ một hàm

**Câu 29.** [L.O.2.1] Cho đoạn mã trong Python như sau:

```
1 def foo(f, x): return f(x)
2 foo(lambda a: a ** 2, 4)
```

Đoạn mã nào trong ngôn ngữ lập trình C++ sau đây tương đương với đoạn mã Python trên:

- ☐ (A) 

```
int foo(int (*f)(int), int x) { return f(x); }
int main() {
    auto lambda = int f(int a) { return a * a; };
    foo(lambda, 4);
    return 0;
}
```
- ☐ (B) 

```
int foo(int *f(int), int x) { return f(x); }
int main() {
    auto lambda = int f(int a) { return a * a; };
    foo(lambda, 4);
    return 0;
}
```
- ☒ (C) 

```
int foo(int (*f)(int), int x) { return f(x); }
int main() {
    auto lambda = [](int a) { return a * a; };
    foo(lambda, 4);
    return 0;
}
```**
- ☐ (D) 

```
int foo(int *f(int), int x) { return f(x); }
int f(int a) { return a * a; };
int main() {
    foo(f, 4);
    return 0;
}
```

**Câu 30.** [L.O.2.1] Có bao nhiêu hàm bậc cao (high-order function) trong số các hàm thư viện sau: `max`, `filter`, `round`, `abs`, `map`?

- ☐ (A) 1      **☒ (B) 2**      ☐ (C) 3      ☐ (D) 4



**Câu 31.** [L.O.2.1] Cho hàm `is_matrix` được định nghĩa trong Python như sau:

```
1 def is_matrix(matrix):
2     if not matrix: return False
3     fl = len(matrix[0])
4     return reduce(lambda x, y: x or len(y) == fl, matrix, True)
```

Đoạn mã nào cần điền vào khoảng trống ở dòng 4 để hàm trên trả về True khi `matrix` là một ma trận, ngược lại trả về False:

(A) `lambda x, y: x or len(y) == fl`

(B) `lambda x, y: fl and len(y) == x`

(C) `lambda x, y: x and len(y) == fl`

(D) `lambda x, y: x and len(y) == len(matrix)`

**Câu 32.** [L.O.2.1] Các ngôn ngữ lập trình hàm thuần khiết không có các cấu trúc lặp dựa trên biểu thức luận lý như phát biểu `while`, do `while` trên C vì trên ngôn ngữ lập trình hàm

(A) không thể thay đổi giá trị của biến

(B) không có biểu thức luận lý

(C) có hàm bậc cao thay thế

(D) dùng đệ quy để thay thế

**Câu 33.** [L.O.2.1] Sau khi hoàn thành (lập trình và kiểm thử) hàm `funcA` viết trên Python để thực hiện một chức năng trên web, bạn muốn điều chỉnh để chức năng này chỉ được sử dụng sau khi người dùng đăng nhập. Một giải pháp đơn giản là thêm `@login_required` vào trước dòng khai báo hàm `funcA`. Trên Python, `login_required` là

(A) một tổ chức dữ liệu

(B) một lớp trong thư viện

(C) một cấu trúc điều khiển

(D) một hàm bậc cao

nhận vào 1 hàm và trả về kq

**Câu 34.** [L.O.2.1] Hãy cho biết kết quả xuất ra màn hình của đoạn mã (trong ngôn ngữ Python) sau:

```
1 def square(x): return x ** 2
2 def double(x): return x * 2
3 numbers = [1, 2, 3, 4, 5]
4 result = map(double, filter(lambda x: x % 2 == 0, map(square, numbers)))
5 print(list(result))
```

(A) [4, 16]

(B) [4, 16, 36, 64, 100]

(C) [2, 18, 50]

(D) [8, 32]

**Câu 35.** [L.O.2.1] Hàm hợp của các hàm  $f_1, f_2, \dots, f_n$  là hàm  $h$  sao cho  $h(x) = f_n(\dots(f_2(f_1(x))))$ . Hãy điền vào chỗ trống trong đoạn mã sau để hàm `compose` sẽ trả về hàm hợp của các thông số của nó?

```
1 from functools import reduce
2 def compose(*f):
3     def inner(x):
4         return reduce(lambda a,b: _____, f, x)
5     return inner
```

(A) `a(b),f`

(B) `b(a),f[::-1]`

(C) `b(a),f`

(D) `a(b),f[::-1]`

Phần giới thiệu sau áp dụng cho các câu hỏi 36–45:

Cho đoạn ngữ pháp được viết trong ANTLR4 cho ngôn ngữ BMirror như sau:

```
program: assign_stmt* EOF;
assign_stmt: assign NEW_LINE;
assign: ID CM assign CM exp | ID EQ exp;
exp: exp (PLUS | MINUS) term | term;
term: term (MUL | DIV) fact | fact;
fact: ID | INTLIT | idx_op;
idx_op: ID (LB INTLIT RB)+;
ID: [a-z]+; CM: ','; SM: ';'; EQ: '=';
INTLIT: [0-9]+; LB: '['; RB: ']'; NEW_LINE: '\r'? '\n';
PLUS: '+'; MINUS: '-'; MUL: '*'; DIV: '/';
WS: [ \t] -> skip;
```

và các lớp AST được khai báo trong ngôn ngữ Python3 như sau:

```
class AST
```

```

class Program(AST): # stmts: List[Stmt]
class Stmt(AST)
class Assign(Stmt): # lhs: Id, right: Exp
class Exp(AST)
class BinExp(Exp): # op: str, left: Exp, right: Exp
class IdxOp(Exp): # base: Exp, idx: int
class IntLit(Exp): # value: int
class Id(Exp): # name: str

```

Với chuỗi nhập viết trên ngôn ngữ BMirror như sau:

```

a, b, c = 1 + 2, a * 2, 4 / b
d = arr[1][2][3]

```

AST tương ứng cần phải được sinh ra như sau:

```

Program([
    Assign(Id("a"), BinExp("+", IntLit(1), IntLit(2))),
    Assign(Id("b"), BinExp("*", Id("a"), IntLit(2))),
    Assign(Id("c"), BinExp("/", IntLit(4), Id("b"))),
    Assign(Id("d"), IdxOp(IdxOp(IdxOp(Id("arr"), 1), 2), 3))
])

```

Đoạn mã sau với một số chỗ trống và một số đoạn bị che (*## Hidden code*) được sử dụng để thực hiện việc sinh AST từ câu cú pháp được tạo ra bởi bộ phân tích cú pháp do ANTLR sinh ra trên Python3 là

```

1 class ASTGenerator(BMirrorVisitor):
2     def visitProgram(self, ctx):
3         assign_stmts = []
4         for assign in ctx.assign_stmt(): assign_stmts += self.visit(assign)
5         return Program(assign_stmts)
6     def visitAssign_stmt(self, ctx):
7         ids, explist = self.visit(ctx.assign())
8         return _____(1)_____
9     def visitAssign(self, ctx):
10        if ctx.getChildCount() == 3: ID EQ exp
11            return _____(2)_____
12        ids, explist = self.visit(ctx.assign())
13        return [Id(ctx.ID().getText())] + _____(3)_____ + [self.visit(ctx.exp())]
14    def visitExp(self, ctx):
15        ## Hidden Code
16    def visitTerm(self, ctx):
17        ## Hidden Code
18    def visitFact(self, ctx): fact: ID | INTLIT | idx_op;
19        if ctx.ID(): return Id(ctx.ID().getText())
20        elif ctx.INTLIT(): return IntLit(int(ctx.INTLIT().getText()))
21        return _____(4)_____
22    def visitIdx_op(self, ctx):
23        return reduce(lambda x,y:_____ (5)_____, _____(6)_____, _____(7)_____)

```

```

program: assign_stmt* EOF;
assign_stmt: assign NEW_
LINE;
assign: ID CM assign CM exp |
ID EQ exp; exp: exp (PLUS |
MINUS) term | term; trái
term: term (MUL | DIV)
fact | fact; fact: ID |
INTLIT | idx_op; idx_op:
ID (LB INTLIT RB)+;
ID: [a-z]+; CM: ','; SM: ':';
EQ: '=';
INTLIT: [0-9]+; LB: '['; RB:
']'; NEW_LINE: '\r'? '\n';
PLUS: '+'; MINUS: '-'; MUL:
'*'; DIV: '/';
WS: [ \t] -> skip;

```

**Câu 36.** [A2] Kiểu đầy đủ của tham số ctx trong phương thức visitAssign\_stmt là

- (A) BMirrorParser.Assign\_stmtContext (B) BMirrorParserTree.Assign\_stmtContext  
(C) BMirror.Assign\_stmtContext (D) BMirrorParser.Assign\_stmt

**Câu 37.** [A2] Chỗ trống (7) ở dòng số 23 nên là

- (A) , Id(ctx.ID().getText()) (B) , []  
(C) , ctx.ID() (D) Không cần mã cho chỗ trống này

**Câu 38.** [A2] Chỗ trống (6) ở dòng số 23 nên là

- (A) ctx.ID() (B) int(ctx.INTLIT().getText())  
(C) self.visit(ctx.INTLIT()) (D) ctx.INTLIT()

số 6 cần 1 danh sách

**Câu 39.** [A2] Chỗ trống (5) ở dòng số 23 nên là

- (A) IdxOp(y, x) (B) IdxOp(x, int(y))  
(C) IdxOp(x, int(y.getText())) (D) IdxOp(y, int(x))

**Câu 40.** [A2] Chỗ trống (4) ở dòng số 21 nên là

- (A) `[self.visit(ctx.idx_op())]` (B) `IdxOp(self.visit(ctx.idx_op()))`  
(C) `self.visit(ctx.idx_op())` (D) `ctx.idx_op()`

**Câu 41.** [A2] Nhận định nào sau đây là đúng?

- (A) Số loại đối tượng mà `visitExp` và `visitFact` có thể trả về là giống nhau và là 4. *fact có 3, exp có 4: BinExpr, ID, INTLIT, idxop*  
(B) `visitFact` có thể trả về đối tượng `Id` trong khi `visitTerm` thì chỉ có thể trả về đối tượng `BinExpr`  
(C) `visitFact` có thể trả về đối tượng `IntLit` trong khi `visitExp` thì chỉ có thể trả về đối tượng `Exp`  
(D) Số loại đối tượng mà `visitExp` và `visitTerm` có thể trả về là 1 và là `BinExpr`.

E

**Câu 42.** [A2] Chỗ trống (2) ở dòng số 11 nên là

- (A) `Assign(Id(ctx.ID().getText()), self.visit(ctx.exp()))`  
(B) `[Id(ctx.ID().getText()), [self.visit(ctx.exp())]`  
(C) `Assign(Id(ctx.ID().getText()), ctx.exp())`  
(D) `(Id(ctx.ID().getText()), self.visit(ctx.exp()))`

`visitFact` có thể trả về `Id` là đúng, nhưng `visitTerm` không chỉ trả về `BinExpr`, mà còn có thể trả về `Id`, `IntLit`, `IdxOp`.

`visitExp` không chỉ trả về `Exp` (lớp cha), mà còn có thể trả về các lớp con của `Exp` như `IntLit`, `Id`, `IdxOp`, `BinExpr`

**Câu 43.** [A2] Chỗ trống (3) ở dòng số 13 nên là

- (A) `self.visit(ctx.assign())` (B) `ids, explist`  
(C) `ids+explist` (D) `ctx.assign()`

**Câu 44.** [A2] Biết rằng `zip` là hàm để tạo một danh sách các phần tử tuple được lấy từ các phần tử có chỉ số tương ứng của các danh sách đầu vào. Chỗ trống (1) ở dòng số 8 nên là

- (A) `[Assign(idc, exp) for idc, exp in list(zip(ids[:-1], explist))]`  
(B) `[Assign(idc, exp) for idc, exp in list(zip(ids[:-1], explist[:-1]))]`  
(C) `[Assign(idc, exp) for idc, exp in list(zip(ids, explist[:-1]))]`  
(D) `[Assign(idc, exp) for idc, exp in list(zip(ids, explist))]`

**Câu 45.** [A2] Chọn lệnh phù hợp để thay thế các dòng lệnh từ 3 đến 5?

- (A) `return Program(reduce(lambda prev, curr: prev + self.visit(curr), ctx.assign_stmt()))`  
(B) `return Program(reduce(lambda prev, curr: prev + [self.visit(curr)], ctx.assign_stmt()))`  
(C) `return Program(reduce(lambda prev, curr: prev + self.visit(curr), ctx.assign_stmt(), []))`  
(D) `return Program(reduce(lambda prev, curr: prev + [self.visit(curr)], ctx.assign_stmt(), []))`

Hết