Lecturer:	(Date)	Approved by:	(Date)
(Signature and Fullname)		(Signature and Fullname)	

	FINAL EXAM	Semester / Academic year	1	2020-2021
UNIVERSITY OF TECHNOLOGY - VNUHCM **FACULTY OF CSE**		Date		13-01-2021
		Course title	Principles Of Programming Languages	
		Course ID	CO3005	
		Duration	120 mins	Question sheet code	1201

Notes: - Students do not use any course materials.
   - Submit the question sheet together with the answer sheet.

I. Programming Part (also used to calculate Assignment 3 and 4): (4 points)

1. (2 points) [LO.3.1] Given the AST classes declared as follows:

```
class AST(ABC)
class Decl(AST)
class Type(AST)
class Program(AST) #decls:list(Decl)
class TypeDecl(Decl): #name:str, rhs:Type
class VarDecl(Decl): # name:str, rhs:Type
class StructType(Type): #ele:list(Decl)
class IntType(Type)
class FloatType(Type)
class Id(Type): #name:str
```

For example, a valid AST is as follows:

```
Program([VarDecl("a",IntType()),
        VarDecl("b",FloatType()),
        TypeDecl("vd",StructType([
                    VarDecl("a",IntType()),
                    VarDecl("b",IntType()),
                    VarDecl("d",FloatType())])),
        VarDecl("c",Id("vd"))])
```

Write class StaticCheck as a Visitor to perform the following tasks:

(a) **Check if a type defined or not**: If an identifier is used as a type in a variable declaration, the identifier must be declared by a type declaration before the variable declaration in the same scope. The exception UndeclareType(name) should be raised otherwise. For example, the last declaration (variable declaration of c) using type vd, which is declared in the third declaration, is valid. If the declaration appeared before the type declaration of vd, the exception UndeclareType("vd") would be raised.

(b) **Return the total of the size of all variables in program**: Let the size of an IntType variable be 2, a FloatType variable be 6 and a StructType variable be the total of the sizes of its members. For example, the size of the above example is 18 (a's size is 2, b's size is 6 and c's size is 10)

2. (2 points) [LO.3.2] Write method **visitUpFor(ast:UpFor,o:Object)** in class CodeGeneration to generate code for statement **upfor**. Given that the class of statement **UpFor** in AST is declared as follows:

```
class UpFor(Stmt): #id:Id, expr1:Expr, expr2:Expr, body:list(Stmt)
```

The meaning of **upfor** statement is given in the following chart.
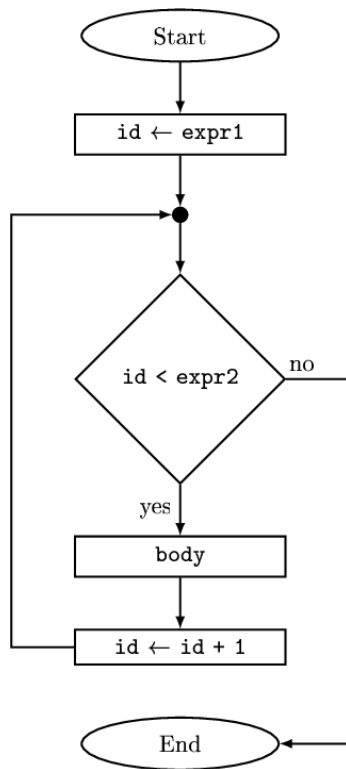
Some following methods of Emitter can be used:

- emitIFTRUE(self, label: int, frame)
- emitIFFALSE(self, label: int, frame)
- emitGOTO(self, label: int, frame)
- emitLABEL(self, label: int, frame)
- emitREADVAR(self,name,intype,index,frame)
- emitWRITEVAR(self,name,intype,index,frame)
- emitPUSHICONST(self,in:int,frame)
- emitADDOP(self,op:str,intype,frame)
- emitREOP(self,op:str,intype,frame)
- prinout(self,out:str)

- emitVAR(self,idx:int,name,intype,fromLabel,toLabel,frame)

Some methods of Frame can be used:

- enterLoop(self)
- exitLoop(self)
- enterScope(self)
- exitScope(self)
- getNewIndex(self)
- getNewLabel(self)
- getBreakLabel(self)
- getContinueLabel(self)

Make sure that the labels for **break** and **continue** statement must be placed in suitable positions. Note that the variable id and expr2 of the **upfor** statement are protected.

II. Written Part:(6 points)

3. (2 points) [LO.2] Given a fragment write in a C-like language as follows:

```
int A[3] = {6,9,15};// index of A starts from 0
int j = 0;
int n = 3;
int sumAndIncrease(int a, int i) {
    int s = 0;
    for ( ; i < n; i = i + 1) {
        s = s + a;
        A[j] = A[j] + 1;
    }
    return s;
}
void main(){
    int s = sumAndIncrease(A[j],j);
    cout << s << A[0] << A[1] << A[2]; //1
}
```

a) 18 9 9 15

b) 15 8 7 15

c) 21 6 8 16

d) 27 6 8 16

Identify and explain the printed results in the following cases:

(a) if a and i are passed by **value**

(b) if a and i are passed by **value-result**.

(c) if a and i are passed by **reference**.

(d) if a and i are passed by **name**.

4. (2 points) [LO.2]

(a) Explain what do you know about exception? What is the difference between exception and simple call-return?

(b) Given the following Python 3 fragment code:

```python
def f():
    x = 1
    try:
        x = 2
        raise Exception('Test')
        x = 3
    except:
        return x
    finally:
        x = 4
        print(x)
    return x
print(f())
```

Present the execution of the above fragment code to explain the printed results?

5. (2 points) [LO.2] Given the following code written in C++:

```cpp
#include <iostream>
int* foo(int x) {
    int *s = new int;
    *s = 3;
    switch (x) {
        case 3: return &x;
        case 4: return foo(*s);
        default: return foo(x-1);
    }
}
int main() {
    int * x = foo(5);
    std::cout << *x;
}
```

(a) Draw the activation records of functions **main** and **foo** until foo is called with actual parameter 3?

(b) Identify and explain some possible errors (dangling reference, garbage) when the above program runs?

————————————————End————————————————