Programming language (2021 1학기)

<HW 4: Call graph>

1.개요

Call graph는 코드 내의 함수들 사이의 호출을 시각화하는 그래프이다. 렉스와 야크를 이용해 코드를 파싱한 후에 함수 호출과 관련한 내용(Ex. 어떤함수가 어떤 함수를 호출하는지, 몇 번 호출하는지, 코드의 몇 번째 줄에서 호출하는지 등)을 자료구조에 저장한다. 그 후 콜 그래프 형태로 출력하는 프로그램을 작성한다.

2.순서

(1) yacc program을 실행시 C코드를 읽어오게 된다.
./hw4 < test.c
(2) 코드의 내용을 분석한다. (test.c 파일의 내용을 읽어온다.)
(3) 코드 내에서의 함수 호출 관계를 파악하고 자료구조에 저장한다.
(4) 자료구조에 저장된 내용을 콜 그래프의 형태로 출력한다.
(예시) 예시일뿐 이번 과제에 참고만 한다.

```
void a(){}
void b(){a();}
void c(){}
void d(){b();}
void e(){}
void g(){e();}
void f(){
  int i;
  for(i=0 ; i<3 ; i++){
    switch(i){
      case 1:  b(); break;
      case 2:  d(); break;
      case 3:  c(); break;
      default: g(); break;
    }
  }
}

void run(){
  srand(time(NULL));
  while(1){
    int ranValue = rand();
    int sel = ranValue % 16;
    switch(sel){
      case 1:  a(); break;
      case 2:  b(); break;
      case 3:  c(); break;
      case 4:  b(); d(); break;
      case 5:  e(); break;
      default: f(); break;
    }
    if(sel == 11)
      break;
  }
}
```

그림 1. test 파일 예시

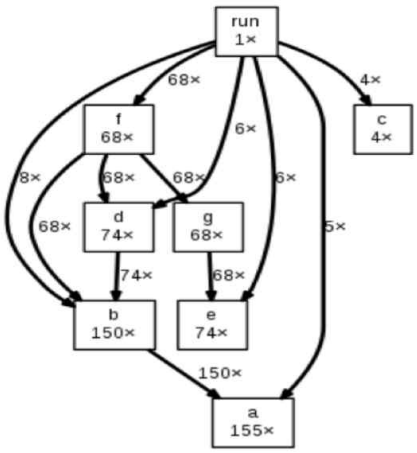
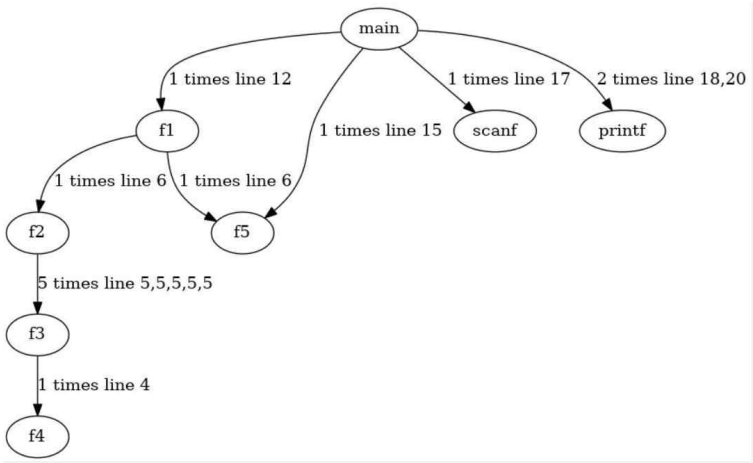


그림 2. call graph 예시

3. 요구사항

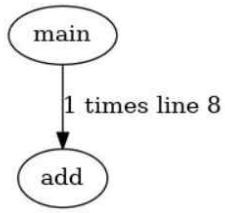
출력 형태는 .jpg(실습시간 dot 사용, 그래프 이미지 출력)으로 고정. 출력 파일 이름은 학번으로 한다.(B000000.jpg).
그래프에는 '함수 호출 횟수', '몇 번째 줄에서 호출하는가?' 반드시 추가.



테스트 케이스 주의사항

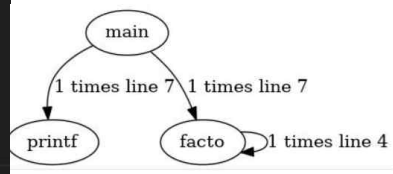
1. 모든 콜 그래프의 루트는 main 함수로 시작합니다. 즉, main부터 실행되어 프로그램 종료까지 호출되지 않는 함수는 그래프에 나타나지 않습니다. 위의 그림처럼 main이 최상위 루트여야합니다.
   Ex)아래와 같을 때

```c
#include<stdio.h>
int add(int a, int b){
    return a+b;
}
void notCalled(){}
int main(){
    add(1,2);
    return 0;
}
```



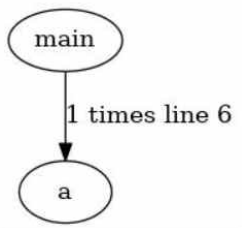
2. 재귀 함수 처리도 가능해야 합니다. 다음 그림과 같이 처리합니다.

```c
#include<stdio.h>
int facto(int n){
    if(n==1){return 1;}
    return n*facto(n-1);
}
int main(){
    printf("%d\n",facto(6));
}
```



3. 반복문을 통해 함수를 호출하는 경우 반복횟수 적용하지 않습니다.

```c
#include<stdio.h>
int a(){}
int main(){
    int i;
    for(i=0;i<10;i++){
        a();
    }
}
```



4. void 자료형이 생략된 함수는 테스트케이스로 주어지지 않습니다.
   ex)void a(){}와 같이 void 자료형은 a(){}로 생략가능합니다. 하지만 테스트케이스에서는 자료형을 명확히 합니다.

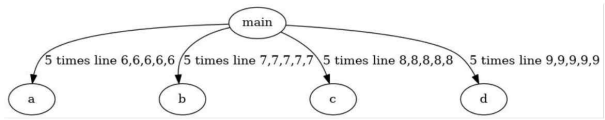
5. yacc과제에서 요구하던 대로 c언어를 파싱하면 됩니다.
   * #define
   * auto,break, case, const, continue, default, double, enum, extern, float, goto, long, register ...
   * 구조체를 반환하는 함수 없음
   * 변수 값은 10진수, 16진수, 8진수...
   등등...
   * typedef에 관한 테스트케이스는 없습니다.

6. 함수를 오버로딩 하지 않습니다.(오버로딩은 c++문법)

7. 모든 테스트케이스는 정상 컴파일 되는 c 소스코드로 진행합니다.

8. 함수의 이름은 최대 20자이며, 함수에서 같은 이름의 함수는 최대 5번 호출됩니다.

```
void a(){}
void b(){}
void c(){}
void d(){}
int main(){
a();a();a();a();a();
b();b();b();b();b();
c();c();c();c();c();
d();d();d();d();d();
}
```



9. 무한 호출하는 테스트케이스는 없습니다.
   ex) void a(){b();}
   void b(){a();}
   int main(){a();return 0;}
   a->b, b->a로 무한 호출하는데 이런 경우는 테스트케이스에없습니다.

10. 그래프의 최대 깊이는 5입니다. 아래 예시는 깊이가 6이 되므로 테스트케이스에 없습니다.

```
void a(){
b();
}
void b(){
c();
}
void c(){
d();
}
void d(){e();
}
void e(){
f();
}
void f(){}
int main(){
a();
return 0;
}
```

11. 지역함수 (함수 내부에서 또다른 함수가 있는 경우) 고려 안하셔도 됩니다.

보너스 과제) 함수포인터
함수 포인터를 이용해 함수를 호출하는 경우가 있습니다. 함수 포인터까지 고려한다면 추가점수 있습니다.

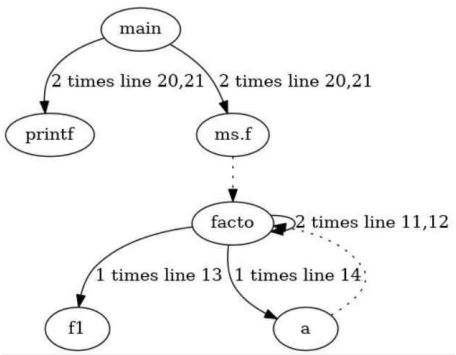
1. 함수포인터 대입을 여러번하는 경우는 고려하지 않습니다.
   ex) a = b = c;

2. 함수의 매개변수, 반환형으로 함수 포인터가 오는 경우는 고려하지 않습니다.

3. 지역,전역 함수 포인터가 있을 수 있습니다.

4. 구조체 멤버로 함수 포인터가 있을 수 있습니다. 해당멤버는 접근 연산자(. , ->)로 호출할 수 있습니다.(테스트케이스 참고).
5. 구조체는 최대 5개까지 올 수 있고, 구조체 이름은 최대 20자입니다. 구조체 멤버로 함수 포인터는 최대 5개까지 올 수 있습니다.

6. 함수 포인터로 함수를 호출할 경우 아래 그림처럼 점선으로 표시합니다.
   (테스트케이스 참고)



4. 문서화

콜 그래프의 개념과 야크에서 구현한 방법에 대한 설명. 함수 호출 관계에 관한 내용을 저장한 자료구조에 관한 설명.
(인터넷이나 책을 참고할 경우 책 제목이나 url 기입).
문서 자체의 비중이 코드의 비중보다 더욱 큼을 유념하도록 합니다.

5.제출

hw4.y, hw4.l, hw4.tex, hw4.pdf, tex에 첨부한 이미지파일(존재한다면)

(1) 보고서 내용
 1. 콜 그래프의 개념과 야크에서 구현한 방법에 대한 설명(동작방식) 자료구조에 관한설명 포함.
 2. 콜 그래프 구현한 야크 코드에서 핵심적인 부분에 관한 자세한 코드 설명. 보고서 내에 코드를 추가할 것.(verbatim 패키지 이용 추천)

(2) 제출 마감 시간 및 장소
 과제기간: 2주
 1. 제출 마감 시간 : 5월 7일 00시(1,2분반), 5월 8일 00시(3,4분반)
 2. 명령어 : submit pem_ta hw4_
 (_ == 1분반: a ,2분반: b, 3분반: c, 4분반: d)

6. 유의사항

(1) 채점 기준

1. 어느 정도 이해하였는가(문서작성을 보고 파악)?
2. 프로그램의 구동은 잘 되는가?

(2) 감점 사항

부정행위 발견 시 관련 학생 모두 0점 처리합니다. 부정행위 의심학생을 특정지어서 학기말에 면담 후, 과제 구현에 대한 이해 부족이 드러나면 부정행위로 간주하여 0점 처리합니다. 과목은 F처리 됩니다.

(3) 질문사항

조교 이메일 : (pemta818@gmail.com)

3-1) 수업시간에 여러번 알려 주었거나 강의록에 명시되어 있는 부분은 답변하지 않음.

3-2) 간단한 구글링을 통해 해답을 얻을 수 있는 질문은 답변하지 않음.