

15-213, 20xx년 가을

데이터 랩: 비트 조작

할당됨 8월 30일, 마감일: 9월 12일(수) 오후 11시 59분

해리 보빅(bovik@cs.cmu.edu) 이 이 과제의 책임자입니다.

1 소개

이 과제의 목적은 정수와 부동 소수점 숫자의 비트 수준 표현에 더 익숙해지는 것입니다. 이를 위해 일련의 프로그래밍 "퍼즐"을 풀어야 합니다. 퍼즐의 대부분은 상당히 인위적인 것이지만, 퍼즐을 풀다 보면 비트에 대해 훨씬 더 많이 생각하게 될 것입니다.

2 물류

이것은 개별 프로젝트입니다. 모든 과제는 전자식으로 제출됩니다. 설명 및 수정 사항은 코스 웹 페이지에 게시됩니다.

3 유인물 지침

사이트 특징: 여기에 교수자가 학생에게 `datalab-handout.tar` 파일을 배포하는 방법을 설명하는 단락을 삽입합니다.

먼저 작업을 수행할 Linux 컴퓨터의 (보호된) 디렉터리에 `datalab-handout.tar`를 복사합니다. 그런 다음 다음 명령을 실행합니다.

```
unix> tar xvf datalab-handout.tar.
```

이렇게 하면 디렉터리에서 여러 파일이 압축이 풀리게 됩니다. 수정하여 제출할 파일은 `bits.c`뿐입니다.

bits.c 파일에는 13개의 프로그래밍 퍼즐 각각에 대한 골격이 포함되어 있습니다. 여러분의 과제는 정수 퍼즐의 직선 코드(즉, 루프나 연산자 없이)와 제한된 수의 C 산술 및 논리 연산자만을 사용하여 각 함수 골격을 완성하는 것입니다. 구체적으로 다음 8가지 연산자만 사용할 수 있습니다:

! ~ & ^ | + << >>

일부 함수는 이 목록을 더욱 제한합니다. 또한 8비트보다 긴 상수는 사용할 수 없습니다. 자세한 규칙과 원하는 코딩 스타일에 대한 논의는 bits.c의 주석을 참조하세요.

4 퍼즐

이 섹션에서는 bits.c에서 풀게 될 퍼즐에 대해 설명합니다.

표 1에는 가장 쉬운 것부터 가장 어려운 것까지 난이도 순으로 퍼즐이 나열되어 있습니다. "Rating" 필드에는 퍼즐의 난이도(점수)가 표시되어 있고, "Max ops" 필드에는 각 함수를 구현하는 데 사용할 수 있는 최대 연산자 수가 나와 있습니다. 함수의 원하는 동작에 대한 자세한 내용은 bits.c의 주석을 참조하세요. 함수의 코딩 규칙을 충족하지는 않지만 함수의 올바른 동작을 표현하기 위한 참조 함수로 사용되는 테스트 함수는 tests.c에서 참조할 수도 있습니다.

이름	설명	평가	최대 운영
bitXor(x, y)	및 및 ~만 사용하여 $x \oplus y$ 를 입력합니다.	1	14
tmin()	가장 작은 2의 보수 정수	1	4
isTmax(x)	x 가 2의 합이 가장 큰 정수인 경우에만 참입니다.	1	10
모든 홀수 비트(x)	x 의 모든 홀수 비트가 1로 설정된 경우에만 참입니다.	2	12
부정(x)	연산자를 사용하여 $-x$ 를 반환합니다.	2	5
isAsciiDigit(x)	$0x30 \leq x \leq 0x39$ 이면 참입니다.	3	15
조건부	$x \neq y$: z 와 동일	3	16
isLessOrEqual(x, y)	$x \leq y$ 이면 참, 그렇지 않으면 거짓	3	24
logicalNeg(x)	연산자를 사용하지 않고 $!x$ 를 계산합니다.	4	12
howManyBits(x)	2의 연산에서 x 를 표현하기 위한 최소 비트 수입니다.	4	90
floatScale2(uf)	$f.p.$ 인수에 대해 $2 * f$ 의 비트 수준 증가물을 반환합니다.	4	30
floatFloat2Int(uf)	f 에 대해 (int) f 의 비트 수준 증가값을 반환합니다. 인수 f .	4	30
floatPower2(x)	정수 x 에 대해 2.0^x 에 해당하는 비트 레벨을 반환합니다.	4	30

표 1: 데이터랩 퍼즐. 부동 소수점 퍼즐의 경우, 값 f 는 부호 없는 정수 uf 와 동일한 비트 표현을 갖는 부동 소수점 숫자입니다.

부동소수점 퍼즐에서는 몇 가지 일반적인 단정밀도 부동소수점 연산을 구현하게 됩니다. 이러한 퍼즐에서는 표준 제어 구조(조건부, 루프)를 사용할 수 있으며, 임의의 부호 없는 정수 상수를 포함하여 정수와 부호 없는 데이터 유형을 모두 사용할 수 있습니다. 유니온, 구조체 또는 배열은 사용할 수 없습니다. 가장 중요한 것은 부동소수점 데이터 유형, 연산 또는 상수를 사용할 수 없다는 것입니다. 대신 모든 부동소수점 피연산자는 함수에 다음과 같은 유형으로 전달됩니다.

를 반환하며, 반환되는 부동 소수점 값은 부호 없는 유형이 됩니다. 코드는 지정된 부동 소수점 연산을 구현하는 비트 조작을 수행해야 합니다.

포함된 프로그램 `fshow`는 부동 소수점 숫자의 구조를 이해하는 데 도움이 됩니다. 컴파일하려면 `fshow`를 클릭하고 유인물 디렉토리로 전환하여 입력합니다:

유닉스> 만들기

`fshow`를 사용하면 임의의 패턴이 부동 소수점 숫자로 표시되는 것을 확인할 수 있습니다:

```
unix> ./fshow 2080374784
```

부동 소수점 값 2.658455992e+36

비트 표현 0x7c000000, 부호 = 0, 지수 = f8, 분수 = 000000 정규화. 1.0000000000 x 2¹²¹

`fshow`에 16진수 및 부동소수점 값을 지정하면 해당 비트 구조를 해독할 수도 있습니다.

5 평가

점수는 다음 분포에 따라 최대 67점 만점으로 계산됩니다:

정답 점수 **36점**.

26 성과 점수.

5 스타일 포인트.

정답 점수. 풀어야 하는 퍼즐은 1에서 4 사이의 난이도 등급이 부여되어 있으며, 가중치 합이 총 36이 되도록 되어 있습니다. 다음 섹션에 설명된 `btest` 프로그램을 사용하여 함수를 평가합니다. 퍼즐이 `btest`에 의해 수행된 모든 테스트를 통과하면 전체 점수를 받게 되며, 그렇지 않으면 점수를 받지 못합니다.

수행 점수. 이 시점에서 가장 중요한 것은 정답을 맞추는 것입니다. 하지만 가능한 한 짧고 단순하게 풀어야 한다는 감각을 심어주고 싶습니다. 또한 일부 퍼즐은 무차별 대입으로 풀 수 있지만, 더 영리하게 풀 수

있기를 바랍니다. 따라서 각 기능에 대해 사용할 수 있는 최대 연산자 수를 설정했습니다. 이 제한은 매우 관대하며 지독하게 비효율적인 솔루션을 잡기 위해 고안되었습니다. 연산자 제한을 만족하는 정답을 맞힐 때마다 2점을 받게 됩니다.

스타일 점수. 마지막으로, 솔루션의 스타일과 댓글에 대한 주관적인 평가를 위해 5점을 배정했습니다. 솔루션은 가능한 한 깔끔하고 간결해야 합니다. 댓글은 유익한 정보를 제공해야 하지만 장황할 필요는 없습니다.

작업물 자동 채점

작업의 정확성을 확인하는 데 도움이 되는 몇 가지 자동 채점 도구(`btest`, `dlc`, `driver.pl`)가 유인물 디렉터리에 포함되어 있습니다.

- **btest**: 이 프로그램은 `bits.c`에 있는 함수의 기능적 정확성을 검사합니다. 이 프로그램을 빌드하여 사용하려면 다음 두 명령을 입력합니다:

```
유닉스> 유닉스 만
```

```
들기> ./btest
```

`bits.c` 파일을 수정할 때마다 `btest`를 다시 빌드해야 한다는 점에 유의하세요.

각 함수를 한 번에 하나씩 테스트하면서 작업하는 것이 도움이 될 것입니다. `f` 플래그를 사용하여 `btest`에 단일 함수만 테스트하도록 지시할 수 있습니다:

```
unix> ./btest -f bitXor
```

옵션 플래그 `-1`, `-2`, `-3`을 사용하여 특정 함수 인수를 전달할 수 있습니다: `unix>`

```
./btest -f bitXor -1 4 -2 5
```

`btest` 프로그램 실행에 대한 설명서는 `README` 파일을 확인하세요.

- **dlc**: 이것은 각 퍼즐의 코딩 규칙을 준수하는지 확인하는 데 사용할 수 있는 MIT CILK 그룹의 ANSI C 컴파일러의 수정 버전입니다. 일반적인 사용법은 다음과 같습니다:

```
UNIX> ./dlc bits.c
```

프로그램은 잘못된 연산자, 너무 많은 연산자, 정수 퍼즐의 직선형이 아닌 코드 등의 문제를 감지하지 않는 한 자동으로 실행됩니다. `e` 스위치로 실행합니다:

```
unix> ./dlc -e bits.c
```

는 각 함수에 사용된 연산자 수를 출력하도록 합니다. `./dlc -help` 입력

를 클릭해 명령줄 옵션 목록을 확인하세요.

- **driver.pl:** `btest` 및 `dlc`를 사용하여 솔루션의 정확성 및 성능 포인트를 계산하는 드라이버 프로그램입니다. 인수가 필요하지 않습니다:

```
unix> ./driver.pl
```

교수자는 `driver.pl`을 사용하여 솔루션을 평가합니다.

6 핸드인 지침

사이트 특징: 여기에 각 학생에게 `bit.c`를 제출하는 방법을 알려주는 텍스트를 삽입합니다.
솔루션 파일을 다운로드하세요.

7 조언

- `<stdio.h>` 헤더 파일은 `dlc`를 혼동하고 직관적이지 않은 오류 메시지를 발생시키므로 `bits.c` 파일에 포함하지 마세요. `<stdio.h>` 헤더를 포함하지 않고도 디버깅을 위해 `bits.c` 파일에 `printf`를 사용할 수 있지만, `gcc`는 무시해도 되는 경고를 출력합니다.
- `dlc` 프로그램은 C++의 경우나 `gcc`에서 적용하는 것보다 더 엄격한 형태의 C 선언을 적용합니다. 특히 모든 선언은 선언이 아닌 문 앞에 블록(중괄호로 묶은 부분)으로 표시되어야 합니다. 예를 들어 다음 코드에 대해 불평합니다:

```
int foo(int x)
{
    int a = x;
    a *3;          /* 선언문이 아닌 문 */ int b = a; /* 에러: 여
기에 선언이 허용되지 않습니다 */
}
```

8 "교수를 이겨라" 콘테스트

재미로 다른 학생 및 강사와 가장 효율적인 퍼즐을 개발하기 위해 경쟁할 수 있는 "교수를 이겨라" 콘테스트를 선택 사항으로 제공합니다. 목표는 가장 적은 수의 연산자를 사용하여 각 데이터 랩 퍼즐을 푸는 것입니다. 각 퍼즐에서 강사의 연산자 수를 맞추거나 이긴 학생이 우승자가 됩니다!

콘테스트에 응모작을 제출하려면 다음과 같이 입력합니다:

```
unix> ./driver.pl -u '닉네임'
```

닉네임은 35자로 제한되며 영숫자, 아포스트로피, 쉼표, 마침표, 대시, 밑줄, 앰퍼샌드가 포함될 수 있습니다

다. 원하는 만큼 자주 제출할 수 있습니다. 가장 최근 제출물은 실시간 점수판에 표시되며, 닉네임으로만 식별할 수 있습니다. 브라우저에서 다음을 가리키면 스코어보드를 볼 수 있습니다.

`http://$서버_이름:$요청_포트`

사이트-특정: `SERVER_NAME` 및 `$REQUESTD_PORT`를 `./contest/Contest.pm` 파일에서 설정한 값으로 바꿉니다.