Homework #01 Answer

데이터사이언스를 위한 컴퓨팅 1 (2022 년도 2 학기, M3239.005500)

Due: 2022 년 9 월 23 일 (금) 23 시 59 분

1. Compilation Process [40pt]

pow.c get 2 floating numbers as the arguments to calculate x^y. The usage of the program is pow x y
(Note: in c or c++, arguments are passed using argc and argv arguments in the main function)

These are answered based on GSDS server.

1.1 Preprocessing [30pt]

a) pow.c includes 3 header files (stdio.h, stdlib.h, math.h), but we didn't write the file. Please find these 3 files. Report the file path. (Hint: use "cpp -v /dev/null" to know the settings in C preprocessor cpp)

```
/usr/include/stdio.h
/usr/include/stdlib.h
/usr/include/math.h
```

각각 3,3,4 점씩 주어졌으며, local 에서 작업한 것에 대한 route 도 정답으로 인정함. (확실치 않은 route 들에 대해서는 partial credit 주어짐)

b) Find a gcc option to run preprocessing and run it with pow.c. In the preprocessed results, please find the function declaration of pow and printf. Please copy & paste them into the report.

```
extern double pow (double __x, double __y) __attribute__ ((__nothrow__ ,
__leaf__));
extern int printf (const char *__restrict __format, ...);
```

각각 5 점씩 주어졌으며, 정답은 아니지만 유사한 경우 각각 2 점씩 주어짐

c) Does preprocessed result have pow and printf implementation? If the implementations are included, please explain the code bri defly. If not, please explain why they are not implemented.

There is no implementation but the prototype. Implementation of the function is included in the executable file during the linking process.

Yes/no 에 대한 답안에 대해서는 5 점, 설명에 대해서 5 점씩 주어졌음.

1.2 Compilation [10pt]

a) Find a gcc options 1) to compile pow.c to produce pow.o and 2) to link pow.o with libraries and produce pow

1) `gcc -c pow.c -o pow.o`
2) `gcc -o pow pow.c`

각각 5 점씩 주어졌음. `gcc pow.c -o pow -lm` 은 compile 과 linking 을 둘다 하는 코드이지만 둘 다 정답으로 인정함.

2. C++ Programming Practice [60pts, NO partial credit]

You can code in many different ways. This is one of the examples.

newbear.h

```
#pragma once
#include <stdlib.h>


class Bear {
       public:
          Bear(float aWeight); // constructor

          void SetWeight(float aWeight); // accessors
          float GetWeight(void);
          virtual float Aggressiveness(void);
          virtual void PrintSelf() = 0;

          protected:
          float weight;
       };


class Polar : public Bear {
       public:
            Polar(float aWeight) : Bear(aWeight) {}; // constructor

            void PrintSelf();
       };

class Grizzly : public Bear {
       public:
            Grizzly(float aWeight) : Bear(aWeight) {}; // constructor

            float Aggressiveness(void);
            void PrintSelf();
       };

class Black : public Bear {
       public:
            Black(float aWeight) : Bear(aWeight) {}; // constructor
```

```
            virtual float Aggressiveness(void);
            void PrintSelf();
       };

class Black_Momma : public Black {
       public:
            Black_Momma(float aWeight) : Black(aWeight) {
               int nCub = 0;
               cub1 = NULL;
              cub2 = NULL;
            }; // constructor

            void AddCub(Black *aCub);
            float Aggressiveness(void);
            float TotalAggressiveness(void);
            void PrintSelf();

       protected:
            Black *cub1;
            Black *cub2;
            int nCub;

       };
```

newbear.cpp

```
#include <stdio.h>
#include "NewBear.h"


/*--------------------- Bear ---------------------*/

/*** Constructor ***/
Bear::Bear(float aWeight) {
   weight = aWeight;
}

/*** Accessors ***/
void Bear::SetWeight(float aWeight) {
   weight = aWeight;
}

float Bear::GetWeight(void) {
   return(weight);
}

float Bear::Aggressiveness(void) {
    return(weight);
}


/*--------------------- Polar ---------------------*/
```

```
void Polar::PrintSelf() {
  printf("I am Polar bear, Weight=%f, Aggressiveness=%f \n", weight,
Aggressiveness());
}

/*-------------------- Grizzly  --------------------*/

float Grizzly::Aggressiveness(void) {
    return(Bear::Aggressiveness() * 0.9);
}

void Grizzly::PrintSelf() {
  printf("I am Grizzly bear, Weight =%f, Aggressiveness=%f \n", weight,
Aggressiveness());
}

/*-------------------- Black --------------------*/

float Black::Aggressiveness(void) {
    return(Bear::Aggressiveness() * 0.7);
}

void Black::PrintSelf() {
  printf("I am Black bear, Weight=%f, Aggressiveness=%f \n", weight,
Aggressiveness());
}

/*-------------------- Black_Momma --------------------*/


void Black_Momma::AddCub(Black *aCub) {
   if (nCub == 0)
       cub1 = aCub;
   else
       cub2 = aCub;
   nCub++;
}

float Black_Momma::Aggressiveness(void) {
    if (nCub == 0){
        return(Black::Aggressiveness());
     }
    else {
       return(Black::Aggressiveness() * 2);
     }
}

float Black_Momma::TotalAggressiveness(void){
   float cubAggressiveness1 = 0;
   float cubAggressiveness2 = 0;

   if (nCub == 0){
```

```
      cubAggressiveness1 = 0;
      cubAggressiveness2 = 0;
   }
   else if (nCub == 1){
     cubAggressiveness1 = cub1->Aggressiveness();
     cubAggressiveness2 = 0;
   }
   else {
     cubAggressiveness1 = cub1->Aggressiveness();
     cubAggressiveness2 = cub2->Aggressiveness();
   }

   return(Aggressiveness() +cubAggressiveness1 + cubAggressiveness2);

}

void Black_Momma::PrintSelf() {
  printf("I am Black Mommabear with %d cub(s), Weight=%f, Aggressiveness=%f, Total
Aggressiveness=%f \n", nCub, weight, Aggressiveness(), TotalAggressiveness());
}
```