#include <assert.h>

#include <limits.h>

#include <math.h>

#include <stdbool.h>

#include <stddef.h>

#include <stdint.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char\* readline();

char\*\* split\_string(char\*);

// Complete the catAndMouse function below.

// Please either make the string static or allocate on the heap. For example,

// static char str[] = "hello world";

// return str;

//

// OR

//

// char\* str = "hello world";

// return str;

//

char\* catAndMouse(int x, int y, int z) {

int dist1,dist2;

dist1 = x-z;

char \*str1 = "Cat A";

char \*str2 = "Cat B";

char \*str3 = "Mouse C";

if(dist1<0)

{

dist1 = dist1 \* -1;

}

dist2 = y-z;

if(dist2<0)

{

dist2 = dist2 \* -1;

}

if(dist1>dist2)

{

//printf("Cat B");

return str2;

}

else if(dist1<dist2)

{

// printf("Cat A");

return str1;

}

else

{

// printf("Mouse C");

return str3;

}

}

int main()

{

FILE\* fptr = fopen(getenv("OUTPUT\_PATH"), "w");

char\* q\_endptr;

char\* q\_str = readline();

int q = strtol(q\_str, &q\_endptr, 10);

if (q\_endptr == q\_str || \*q\_endptr != '\0') { exit(EXIT\_FAILURE); }

for (int q\_itr = 0; q\_itr < q; q\_itr++) {

char\*\* xyz = split\_string(readline());

char\* x\_endptr;

char\* x\_str = xyz[0];

int x = strtol(x\_str, &x\_endptr, 10);

if (x\_endptr == x\_str || \*x\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* y\_endptr;

char\* y\_str = xyz[1];

int y = strtol(y\_str, &y\_endptr, 10);

if (y\_endptr == y\_str || \*y\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* z\_endptr;

char\* z\_str = xyz[2];

int z = strtol(z\_str, &z\_endptr, 10);

if (z\_endptr == z\_str || \*z\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* result = catAndMouse(x, y, z);

fprintf(fptr, "%s\n", result);

}

fclose(fptr);

return 0;

}

char\* readline() {

size\_t alloc\_length = 1024;

size\_t data\_length = 0;

char\* data = malloc(alloc\_length);

while (true) {

char\* cursor = data + data\_length;

char\* line = fgets(cursor, alloc\_length - data\_length, stdin);

if (!line) { break; }

data\_length += strlen(cursor);

if (data\_length < alloc\_length - 1 || data[data\_length - 1] == '\n') { break; }

size\_t new\_length = alloc\_length << 1;

data = realloc(data, new\_length);

if (!data) { break; }

alloc\_length = new\_length;

}

if (data[data\_length - 1] == '\n') {

data[data\_length - 1] = '\0';

}

data = realloc(data, data\_length);

return data;

}

char\*\* split\_string(char\* str) {

char\*\* splits = NULL;

char\* token = strtok(str, " ");

int spaces = 0;

while (token) {

splits = realloc(splits, sizeof(char\*) \* ++spaces);

if (!splits) {

return splits;

}

splits[spaces - 1] = token;

token = strtok(NULL, " ");

}

return splits;

}