```c
#include <stdio.h>

#define MAX 11


enum sym {XX,YY,ORIG};
struct Point {
int id; // int au d?part !
float x;
float y;
};

struct Point droite[MAX];

int symetrique(enum sym option, float *x, float *y);
void AffichePoint(struct Point point);
void LirePoint(int id, float x, float y, struct Point *point);// todo !
int symetrique2(enum sym option, struct Point *point);
void AfficheDroite(struct Point d[]);
void defDroite(struct Point d[],struct Point A, struct Point B);

int main(){

enum sym opt=XX;
struct Point A={0,-3,-40};
// A.id=0; A.x=3; A.y=40;

printf("Symetrique de %d (%.2f,%.2f) selon l'option %d est
",A.id,A.x,A.y,opt);
if (symetrique2(opt,&A))
AffichePoint(A);

//if (symetrique(opt,&A.x,&A.y))
// printf(" %.2f,%.2f\n",p.x,p.y);

struct Point B;
B.id = MAX-1; B.x=5; B.y=12;
defDroite(droite,A,B);

printf("Taille d'une structure point %d ?= %d + %d + %d \n",
sizeof(struct Point),sizeof(A.id),sizeof(A.x),sizeof(A.y));


return 0;
}

/************ Fonctions ************/
void defDroite(struct Point d[],struct Point A, struct Point B){
//Segment

int i;
float alpha, beta, pas;
```

```
alpha=(A.y-B.y)/(A.x-B.x);// Require A.x-B.x != 0
beta = A.y - alpha * A.x;
d[A.id]=A; d[B.id]=B;

pas = (B.x-A.x)/(MAX-1);

//printf("Equation de la droite y = %.2f x + %.2f\n",alpha,beta);
//printf("La pas sur l'axe des X est %.2f\n",pas);

for (i=1;i<MAX-1;i++){
d[i].x = d[i-1].x+pas;
d[i].y = alpha*d[i].x+beta;
d[i].id = d[i-1].id + 1;
}

AfficheDroite(d);

}


int symetrique2(enum sym option, struct Point *point){

switch (option){
case XX :
point->y = -point->y;
return 1;
case YY :
point->x = -point->x;
return 1;
case ORIG :
point->x = -point->x;
point->y = -point->y;
return 1;
default :
printf("\nOption non valide ! ");
return 0;
}

}

//enum sym {XX,YY,ORIG};
int symetrique(enum sym option, float *x, float *y){

switch (option){
case XX :
*y = -*y;
return 1;//break; si pas de return
case YY :
*x = -*x;
return 1;
case ORIG :
*x = -*x;
```

```
*y = -*y;
return 1;
default :
printf("\nOption non valide ! ");
return 0;
}
}


void AffichePoint(struct Point point){
printf("%d (%.2f,%.2f)\n", point.id, point.x, point.y);

}

void AfficheDroite(struct Point d[]){
int i;
for (i=0;i<MAX;i++)
AffichePoint(d[i]);

}
t
```