```
    }
    printf("%d %d %d\n", a, b, c);
    f(c);
    printf("%d %d %d\n", a, b, c);
    return 0;
}
```

Questions

1. Donnez un makefile permettant d'obtenir un exécutable à partir de ces fichiers.
2. Donnez l'affichage produit par cet exécutable.

5 Exercice sur le passage de paramètres

Dans cet exercice, vous allez écrire trois programmes en C réalisant la même opération — une addition de deux entiers par un appel de fonction — qui diffèrent par la méthode de passage de paramètres.

1. Définissez trois variables globales `i`, `j` et `k` de type entier. Écrire une fonction `globadd()` qui fait l'addition de `i` et `j` et stocke le résultat dans `k`. Écrire la fonction `int main()` qui réalise la saisie des variables `i` et `j`, fait appel à la fonction d'addition et retourne le résultat contenu dans `k`.
2. Même exercice que le précédent mais en utilisant le passage de paramètres et le retour de fonction. Les trois variables `i`, `j` et `k` de type entier sont déclarées localement dans la fonction `main()`. La fonction d'addition est une fonction retournant un entier. Ele accepte deux paramètres entiers (`p1` et `p2`) et retourne la somme de ces deux paramètres. La fonction `main()` permet la saisie des deux variables locales `i` et `j` et utilise la fonction d'addition en récupérant le résultat de cette fonction dans la variable locale `k`. Elle retourne le contenu de `k`.
3. Même exercice que le précédent mais en utilisant le passage de paramètres et un pointeur pour modifier une variable dans la fonction appelante. Les trois variables `i`, `j` et `k` de type entier sont declarées localement dans la fonction `main()`. La fonction d'addition `ptadd()` est une fonction qui ne retourne rien. Elle accepte trois paramètres : deux entiers (`p1` et `p2`) et un paramètre de type pointeur vers un entier qui sert à affecter avec la somme de ces deux premiers paramètres, la variable dont la fonction appelante passe l'adresse. La fonction `main()` saisit les deux variables locales `i` et `j` et appelle la fonction d'addition passant l'adresse de la variable locale `k`. Elle retourne le contenu de `k` après l'appel de fonction.