Ex1. Gestion des overflow en OCaml (1h)

On considère le type suivant :

```
type entier = Infini | MoinsInfini | Entier of int;;
```

Je vous informe que `max_int` est le plus grand entier représentable en OCaml et `min_int` le plus petit :

```
# max_int;; (* 2^62 - 1 *)
- : int = 4611686018427387903
# min_int;; (* -2^62 *)
- : int = -4611686018427387904
# max_int + 1;;  (* un dépassement du plus grand entier donne le plus petit *)
- : int = -4611686018427387904
```

On voudrait pouvoir additionner deux entier `a` et `b` en prenant en compte les dépassements de `int` ("integer overflow"), et en donnant `Infini` dans ce cas.

— Q1. Pourquoi ne peut-on pas tout simplement tester si `a + b > max_int` ?

— Q2. Pourquoi peut-on tester à la place si `max_int  a` $\leqslant$ `b`, avec `a`$\geqslant$`0` ?

— Q3. Quel test similaire pourrait-on utiliser pour savoir si `a + b < min_int` ?

— Q4. Écrire une fonction `add_int : int -> int -> entier`, telle que `add_int` ajoute 2 `int` en tenant compte des dépassements (on renvoie `Infini` s'il y a un dépassement de `max_int` par le haut, et `MoinsInfini` s'il y a un dépassement de `min_int` par le bas).

— Q5. Écrire une fonction `add : entier -> entier -> entier` ajoutant 2 `entier` en tenant compte des dépassements. On pourra écrire un `failwith "Forme indéterminée"` dans le cas d'une forme indéterminé.

— Q6. Écrire une fonction `oppose : entier -> entier` renvoyant l'opposé d'un entier. Par exemple, `oppose Infini` renvoie `MoinsInfini`, `oppose (Entier a)` renvoie `Entier (-a)`...

— Q7. Écrire une fonction `sub` effectuant la soustraction. On sera malin et on utilisera les fonctions précédentes.

Pour savoir si `a`$\times$`b` fait un dépassement, on propose la façon suivante :

1. Calculer `c=a`$\times$`b`,
2. Regarder si `c/a` est égal à `b`.

— Q8. Pourquoi cela fonctionne ? À quelle autre erreur faut-il faire attention dans cette méthode ?

Ex2. Petits calculs et tableaux en c (1h)

Le squelette de code suivant est disponible sur `https://cahier-de-prepa.fr/mp2i-kleber/docs?rep=4`, copier-coller le dans Emacs :

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

// tester si C est en appel par valeurs ou par référence pour les types de bases
void incremente_ou_pas(int i) {
    i += 1;
    // si c'est par valeurs, i a été copié et donc le i+=1 ne le modifie pas
    // si c'est par référence, i += 1 changera la variable hors de la fonction
}

double puissance(double x, int n) {
    // TODO code à remplir avec l'exponentiation rapide
    return 1.0;
}

void tableau_puissance(double x) {
    double tableau[20];
    for (int i = 0; i < 20; i++) {
        tableau[i] = puissance(x, i);
        printf("tableau[%i] = %g\n", i, tableau[i]);
    }
    int trop_loin = 21;
    printf("tableau[%i] = %g\n", trop_loin, tableau[trop_loin]);
}

int main(int argc, char* argv[]) {
    int j = 0;
    for (int i = 0; i < argc; i += 1) {
        printf("Argument numéro %i = %s", i, argv[i]);
    }

    int k = j;
    incremente_ou_pas(j);
    printf("incremente_ou_pas(%i) = %i", k, j);
    j += 1; /* ou aussi j = j + 1, mais PAS j++ ou ++j qui sont hors programme */

    tableau_puissance(2.0);

    return EXIT_SUCCESS;  // macro de stdlib pour le code de sortie de réussite = 0
}
```

On rappelle que depuis un terminal, on compile ce fichier avec `COMPILATEUR = gcc` ou `clang` avec la ligne de commande suivante, puis on exécute le binaire produit avec la dernière ligne :


```
$ COMPILATEUR -O0 -Wall -Wextra -Wvla -Werror -fsanitize=address \
              -fsanitize=undefined -o tp6.exe tp6.c -lm
$ ./tp6.exe
```

— Q9. Observer le comportement de la boucle sur `i` de `0` à `argc-1`. Que fait-elle ? Essayer de passer des arguments lors de l'appel à `tp6.exe` ;

— Q10. Observer le comportement de `incremente_ou_pas`. Qu'en déduire quant au comportement de C pour des variables dans les appels de fonctions ? Le langage se utilise-t-il un appel par valeur (en recopiant la valeur) ou par référence (en copiant l'adresse de la valeur) ?

— Q11. Dans la fonction `tableau_puissance`, observez-vous une erreur dans la lecture de la case à l'indice `trop_loin` ? Essayer avec ou sans l'option `-fsanitize=address` et avec ou sans l'option `-fsanitize=undefined`. Qu'est-ce que font ces options ? Pensez-vous qu'il soit utile de les activer toujours, malgré le fait qu'elles ajoutent un surcoût en terme de temps de calcul et de mémoire utilisée ? Vous engagez vous à toujours activer ces options en TP et pour les DM ?

