5.2

Nom et prénom : Wichoud Nicolas

Travail écrit no. 2

(Durée : 2 périodes)

Consignes :
- **ECRIVEZ VOS REPONSES DIRECTEMENT SUR CE DOCUMENT**
- Seules les réponses figurant sur ce document seront corrigées
- Vous pouvez écrire au crayon papier
- Ne pas dégrafer ce document
- Un résumé C figure à la fin de ce document

5

Question 1 (7 points)

Pour chacune des instructions ou suites d'instructions ci-dessous, indiquer dans la colonne de droite du tableau si celle-ci est ***juste*** ou ***fausse***.

IMPORTANT
- Il n'est PAS demandé de justifier vos réponses ou de proposer un quelconque correctif
- Barème appliqué : +0.5 point pour une réponse correcte; 0 point en cas d'absence de réponse; –0.5 point pour une réponse incorrecte

Instructions		Juste ou faux ?
`char chaine[3] = "ABC";`	faux ✓	faux ✓
`char chaine[5];` `chaine = "ABC";`	faux car = n'est plus une initialisation juste ✗	faux ✓

`const char* chaine1;`
`char chaine2[] = "ABC";`
`chaine1 = chaine2;`
	car on parle pas pointeur faux ✗	juste ✓

`typedef char string[];`	juste faux ✗	juste ✓

`struct S {`
`  int n = 1;`
`  double x = 2.5;`
`} s;`
	faux ✓	faux ✓

`typedef struct {`
`  const int N;`
`  double x;`
`} S;`
`S s = {1, 2.5};`
	juste ✓	juste ✓

`struct S {`
`  const int N;`
`  double x;`
`} s;`
`s = (struct S){1, 2.5};`
	faux ✓	~~juste~~

`struct S {`
`  int n;`
`  static int m;`
`};`
`struct S s = {1, 2};`
	faux ✓	faux ✓

`struct S {`
`  int n;`
`  double x;`
`};`
`struct S s1 = {.x = 2.5};`
`struct S s2 = s1;`
	juste ✓	juste ✓

`struct S {`
`  int n;`
`  double x;`
`};`
`struct S s1 = {1, 2.5};`
`struct S s2 = {1, 2.5};`
`printf("s1 == s2 ? %s\n", (s1 == s2 ? "oui" : "non"));`
	faux ✓	faux ✓

`struct S {`
`  enum {A, B, C = 0} e;`
`} s;`
`s.e = B;`
`printf("%d\n", s.e);`
	juste ✓	juste ✓

`struct S1 {`
`  struct S2* a;`
`};`
`struct S2 {`
`  struct S1* a;`
`};`
	juste faux ✗	juste ✓

`typedef struct S {`
`  int n;`
`  T* ptr;`
`} T;`
`T t = {1, NULL};`
	faux ✓	~~juste~~

`union U {`
`  int n;`
`  double x;`
`};`
`union U u = {1.5};`
`printf("u = %f\n", u.x);`
	faux ✓	faux ✓

4.75

Question 2 (5 points)

3

a) **(3 pts)**
Soient les déclarations suivantes :

```
const char* t[] = {"Fiat","Peugeot", "Ford", "Volvo", "Lancia", "Jeep"};
const char** pt[] = {t, t+1, t+2, t+3, t+4, t+5};
```

Quelle valeur fournit chacune des expressions ci-dessous ?
(Conseil : Aidez-vous d'un petit dessin)

1) `*pt - pt[3]`
2) `*pt[2]`
3) `**(*pt+5) - 2`
4) `*t+2`
5) `pt[++pt[3]-*(pt+3)][1][2]`

pt[1][1][2]

IMPORTANT

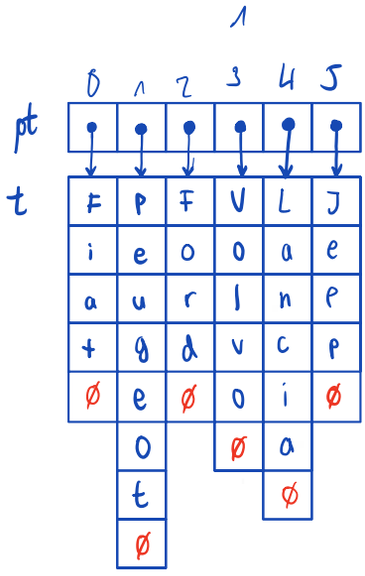
- Les 5 expressions sont supposées être effectuées les unes après les autres, en commençant par la no 1). Le résultat d'une certaine expression peut donc dépendre de ce qui s'est produit lors de l'évaluation des expressions précédentes.

Consigne

- Pour les caractères, donner le résultat entre simples guillemets.
  Exemple : 'a'.
- Pour les chaînes de caractères, donner le résultat entre doubles guillemets.
  Exemple : "ABC".

Vos réponses :

1) -3 ✓ -3
2) "Ford" ✓ "Ford"
3) 72 ✓ 'J'-2
4) "at" ✓ "at"
5) 'u' ✓ 'r'



1.75

b) **(2 pts)**
On suppose disposer d'un programme C, appelé *myprog.c*, dont l'exécutable se nomme *myprog.exe*, et ayant pour entête du main :

```
int main(int argc, char* argv[]) { ... }
```

On suppose maintenant exécuter la ligne de commande suivante :

```
myprog -opt1=10 test.dat 20 -opt2=src
```

Parmi les affirmations suivantes, laquelle (lesquelles) est (sont) vraie(s) ?

A. Dans le code de myprog.c, argc vaut 4
B. Dans le code de myprog.c, argc vaut 5
C. Dans le code de myprog.c, argc vaut 7
D. Dans le code de myprog.c, argv[0] contient la sous-chaîne "main.c"
E. Dans le code de myprog.c, argv[0] contient la sous-chaîne "myprog"
F. Dans le code de myprog.c, argv comporte argc éléments

Votre réponse : B, E et ~~F~~

B, E ✓

5.25

Question 3 (6 points)

3

a) **(3 pts)**
Soit la déclaration suivante :

```
typedef struct {
   unsigned char a;
   uint16_t b;
   uint8_t c;
   size_t d;
   signed char e[2];
   uintptr_t f;
} S;
```

a	x	b	b
c	x	x	x
d	d	d	d
e	e	x	x
f	f	f	f

32bits = 20

a	x	b	b	c	x	x
d	d	d	d	d	d	d
e	e	x	x	x	x	x
f	f	f	f	f	f	f

64bits = 32

Que vaut sizeof(S) en architecture 32 bits ?

Votre réponse : 1+1+2+1+3+4+2+2+4 = 20 ✓

Que vaut sizeof(S) en architecture 64 bits ?

Votre réponse : 1+1+2+1+3+8+2+6+8 = 32 ✓

b) **(3 pts)**

2.25

L'**indice de masse corporelle** ou **IMC** est une grandeur qui permet d'estimer la corpulence (maigre, poids normal, surpoids, etc.) d'une personne. Par exemple, une personne dont l'IMC est compris dans l'intervalle (18.5; 25] a un poids normal, alors qu'une personne dont l'IMC est compris dans l'intervalle (25; 30] est en surpoids. L'IMC d'une personne s'obtient en divisant son poids [kg] par le carré de sa taille [m].
Compléter la partie notée *<à compléter>* dans le code ci-contre (et rien d'autre !) de telle sorte que celui-ci compile sans warnings et affiche à l'exécution :

```
imc[0] = 29
imc[1] = 17
imc[2] = 28.5
```

A noter que le code proposé ci-dessous n'importe ni <math.h>, ni <string.h> !

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

double imc(const char* s) {
   <à compléter>
}

int main(void) {
   const char* const POIDS_ET_TAILLE[] = {"95 1.81", // 95 = poids[kg]
                                                     // 1.81 = taille[m]
                                          "48.5 1.69",
                                          "103 1.9"};
   const size_t NB_ELEM = sizeof(POIDS_ET_TAILLE) / sizeof(char*);
   for (size_t i = 0; i < NB_ELEM; ++i) {
      printf("imc[%" PRIuMAX "] = %.3g\n", i, imc(POIDS_ET_TAILLE[i]));
   }
   return EXIT_SUCCESS;
}
```

```
double imc(const char* s) {
   double masse, taille;
   sscanf("%lf%lf", &masse, &taille);
   return masse / (taille * taille);
}
```

Votre réponse :

Variante avec sscanf

```
double imc(const char* s) {
   double poids, taille;
   sscanf(s, "%lf%lf", &poids, &taille);
   return poids / (taille * taille); // ou poids / taille / taille
}
```

Variante avec strtod

```
double imc(const char* s) {
   char* end;
   const double
      POIDS = strtod(s, &end),
      TAILLE = strtod(end, NULL);
   return POIDS / (TAILLE * TAILLE);
}
```

3.75

Question 4 (6 points)

0

a) **(1.5 pt)**
Que va afficher à l'exécution le programme C ci-dessous :

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

int main(void) {
   char chaine1[] = "XXXXX";
   char chaine2[5];
   const char* chaine3 = "ABC";
   strcpy(chaine2, chaine1);
   strncpy(chaine2, chaine3, strlen(chaine2));
   printf("%s\n", strncpy(chaine1 + 1, chaine3, strlen(chaine2)));
   return EXIT_SUCCESS;
}
```

Votre réponse : ~~XABCX~~

ABCX

b) **(4.5 pts)**

3.75

La fonction *inserer* ci-dessous est censée permettre l'insertion de la chaîne *from* dans la chaîne *to* à la position *pos* :

```
 1 void inserer(char* to, const char* from, const size_t pos) {
 2    if (to && from) {
 3       const size_t STRLEN_TO = strlen(to);
 4       if (pos < STRLEN_TO) {
 5          if (pos == STRLEN_TO) {
 6             strcat(to, from);
 7          } else {
 8             char res[STRLEN_TO + strlen(from)];
 9             if (pos == 0) {
10                strcpy(res, from);
11                strcat(res, to);
12             } else {
13                strncpy(res, to, pos);
14                res[pos + 1] = '\0';
15                strcat(res, from);
16                strcat(res, to);
17             }
18             strcat(to, res);
19          }
20       }
21    }
22 }
```

Cette fonction contient toutefois des maladresses et/ou des erreurs.
Indiquez le no des lignes problématiques et proposez dans chaque cas un correctif (écrit en C).

IMPORTANT

- Une réponse avec no de ligne problématique correct mais sans proposition de correctif sera considérée comme fausse. Idem si le correctif proposé est faux.
- Des points seront décomptés si une réponse, au lieu de corriger une erreur, en ajoute une supplémentaire
- La ligne no 1 ne doit en aucun cas être modifiée
- Aucune ligne de code ne doit être ajoutée, supprimée ou déplacée

Votre réponse :

4) if (pos <= STRLEN_TO) { ✓

8) char res[STRLEN_TO + strlen(from) + 1]; ✓

14) res[pos] = '\0'; ✓

16) strcat(res, to + pos); ✓

18) strcpy(to, res); ✓

Votre réponse (suite) :

Votre réponse :

```
1  void inserer(char* to, const char* from, const size_t pos) {
2     if (to && from && *from) {
3        const size_t STRLEN_TO = strlen(to);
4        if (pos <= STRLEN_TO) {
5           if (pos == STRLEN_TO) { // Si insertion en queue
6              strcat(to, from);
7           } else {
8              char res[STRLEN_TO + strlen(from) + 1];
9              if (pos == 0) { // Si insertion en tête
10                strcpy(res, from);
11                strcat(res, to);
12             } else { // Si insertion ailleurs qu'en tête ou en queue
13                strncpy(res, to, pos);
14                res[pos] = '\0';
15                strcat(res, from);
16                strcat(res, to + pos);
17             }
18             strcpy(to, res);
19          }
20       }
21    }
22 }
```

N.B. La ligne 8 utilise un VLA. C'est autorisé par la norme C, mais pas par la norme C++

10

Question 5 (10 points)

1) **(8 pts)**

8

Ecrire, de la manière la plus propre et la plus modulaire / évolutive possible, toutes les déclarations de constantes et de types (**et rien d'autre !**) permettant de modéliser des vaisseaux spatiaux conformément au cahier des charges suivant :

- Un vaisseau spatial est soit un vaisseau de combat, soit un vaisseau d'exploration.
- Tout vaisseau spatial a un nom (de longueur quelconque) et possède un équipage[1].
  [1] Tout vaisseau comporte un équipage... mais un équipage peut éventuellement être vide
- Un équipage comprend de 0 à 5 membres au maximum.
- Un membre d'équipage se caractérise, pour l'heure, uniquement par son nom (de longueur quelconque).
- Si le vaisseau spatial est un vaisseau de combat, on souhaite enregistrer, en plus de son nom et de son équipage, son poids (nombre entier exprimé en kg) et s'il est équipé ou non de canons laser.
- Si le vaisseau spatial est un vaisseau d'exploration, on souhaite enregistrer, en plus de son nom et de son équipage, son rayon d'action (nombre réel exprimé en milliards de km).

2) **(2 pts)**

2

En supposant le point 1) résolu, déclarer les 2 vaisseaux spatiaux suivants :

- "Starfighter" : un vaisseau de combat de 2500 kg, équipé de canons laser et ayant pour équipage Joe et Jack
- "X-Wing" : un vaisseau d'exploration sans équipage dont le rayon d'action est de 63.2 milliards de km

IMPORTANT
Les déclarations des deux vaisseaux spatiaux doivent être implémentées chacune à l'aide d'**une seule instruction écrite de la manière la plus courte possible**[1].
[1] La présence de toutes les paires d'accolades est toutefois requise.

Vos réponses :

```
#include <stdbool.h>

#define MAX_MEMBRES 5

typedef enum {COMBAT, EXPLO} TypeVaisseau;

typedef struct {
    const char* nom;
} Membre;

typedef struct {
    size_t nbMembres;
    Membre equipage[MAX_MEMBRES];
} Equipage;

typedef struct {
    unsigned poids;   // en kg
    bool canons;
} Combat;
```

Vos réponses (suite) :

```
typedef struct {
    double rayon; // en milliards de km
} Exploration;

typedef union {
    Combat combat;
    Exploration explo;
} Caracteristiques;

typedef struct {
    const char* nom;
    Equipage equipage;
    TypeVaisseau type;
    Caracteristiques caracteristiques;
} Vaisseau;

Vaisseau v1 = {"Starfighter", {2, {{"Joe"}, {"Jack"}}}, COMBAT, {{2500, true}}};
Vaisseau v2 = {"X-Wing", {0, {}}, EXPLO, {.explo = {63.2}}};
```