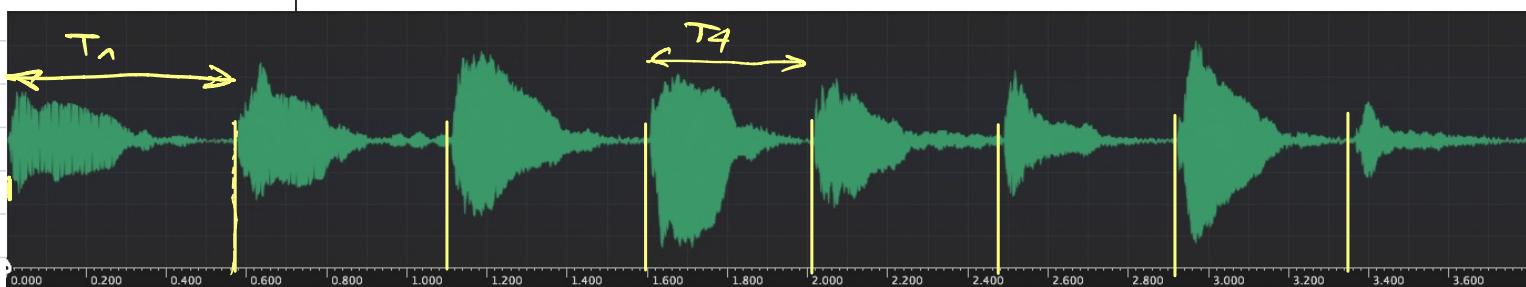


Sinf2 Allocation dynamique

3. 8. 2021



$$T_1 \neq T_2 \neq T_3 \neq T_4 \dots$$

Bon TD20210729 \rightarrow TD20210503



DUREE_MAX_NOTE 0.5
(#distr)
Aligner chaque début de note sur un multiple de 0.5 s.

Si N notes \rightarrow on peut calculer la taille totale du wav final. \rightarrow malloc

Pour les notes individuelles :

Cas 1: le .wav est plus court que DUREE_MAX_NOTE



Cas 2: le .wav est plus long.

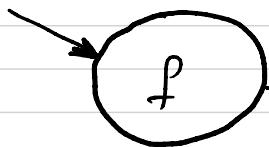


\hookrightarrow mettre dans le wav final

- ① Modifie l'allocation du tableau pour le wav final : t
 - a) calculer le nombre de fichiers wav à lire
 - b) alloue en 1x le tableau t.

Allocation dynamique dans une fonction

fonction



adresse à un tableau alloué dynamiquement dans la fonction
Ce tableau contient les données du .WAV

malloc: réalisé dans la fonction
l'adresse du tableau est un paramètre de sortie.

void f (double *x) {

x: adresse d'un double

*x = 3.14;

*x: contenu du double

}

Si une fonction doit modifier une variable x, on passe son adresse (&x) lors de l'appel.
On utilise *x pour modifier le contenu.

main () {

double z = 0.;

f(&z);

ID: Le double de l'exemple ci-dessus est remplacé par un tableau

main() {

sStereoSample *t = NULL;

f(&t);

void f (sStereoSample *adrTab) {

adrTab = (sStereoSample) malloc (...);

}

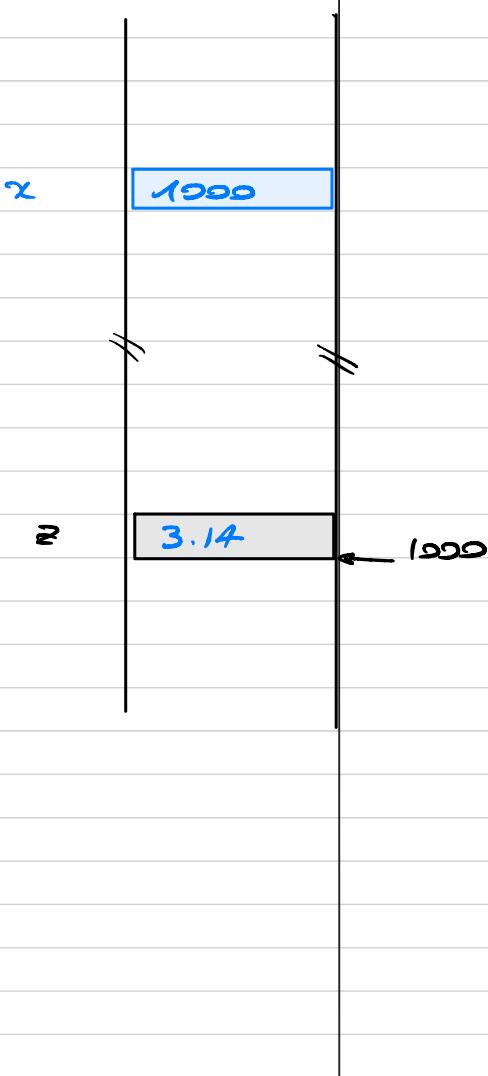
// adrTab: contient l'adresse d'une variable de type tableau de sStereoSample.

```

void f( double *x) {
    *x = 3.14;
}

main() {
    double z = 0.0;
    f(&z);
}

```



```

void f( char* *t ) {

    *t = (char*) malloc (3 * sizeof(char));
}

main() {
    char *s = NULL;
    f( &s );
    // Utilization possible: s[0], s[1], s[2]
}

```



D220503 modifier getWaveData Selon le prototype suivant

```
int getWaveData(char *filename, sStereoSample **area, uint32_t *numSamples);
```

et implementer son contenu

ouvrir fichier / malloc / transfert du wav
→ zone tableau / M.A.J. → numSamples.