Exercice 1 question 1

on réutilise

question 2

Noeud d’arbre

```
typedef struct _noeud_binaire *PNoeudBinaire;
typedef struct _noeud_binaire {
        void *data;
        PNoeudBinaire gauche;
        PNoeudBinaire droit;
} NoeudBinaire;

typedef struct _arbre_binaire *ParbreBinaire;

typedef struct _arbre_binaire *PArbreBinaire;
typedef struct _arbre_binaire {
        char copie;
        PNoeudBinaire racine;
        void *(*dupliquer)(const void *src);
        void (*copier)(const void *src, void *dst);
        void (*detruire)(void *data);
        void (*afficher)(const void *data);
        int (*comparer)(const void *a, const void *b);
        int (*ecrire)(const void *data, FILE *f);
        void *(*lire)(FILE *f);
} ArbreBinaire;
```

question 3

```
PArbreBinaire creer_arbre(char copie,
        void *(*dupliquer)(const void *src),
        void (*copier)(const void *src, void *dst),
        void (*detruire)(void *data),
```

```
        void (*afficher)(const void *data),
        int (*comparer)(const void *a, const void *b),
        int (*ecrire)(const void *data, FILE *f),
        void *(*lire)(FILE *f)
        ) {
        PArbreBinaire pab=(PArbreBinaire)malloc(sizeof(ArbreBinaire));
        if(pab==NULL) {
                printf("Probleme d'allocation de l'arbre binaire.\n");
                return NULL;
        }

        pab->racine=NULL;
        pab->copie=copie;
        pab->dupliquer=dupliquer;
        pab->copier=copier;
        pab->detruire=detruire;
        pab->afficher=afficher;
        pab->comparer=comparer;
        pab->ecrire=ecrire;
        pab->lire=lire;
        return pab;
}
```

question 4 : création d’un noeud

```
PNoeudBinaire creer_noeud_binaire(PArbreBinaire pab, void *data){
        PNoeudBinaire n=(PNoeudBinaire)malloc(sizeof(NoeudBinaire));
        if (n==NULL) {
        printf("Probleme lors de l’allocation d’un noeud binaire.\n");
        return NULL;
        }
if (pab->copie)
        n->data=pab->dupliquer(data);
else{
        n->data=data;
}
n->gauche=NULL;
n->droit=NULL;
return n;
}
```

question 5:

```
void detruire_noeud_binaire(PNoeudBinaire pnb, PArbreBinaire pab) {
        if (pnb) {
        detruire_noeud_binaire(pnb->gauche, pab);
        detruire_noeud_binaire(pnb->droit, pab);
        if (pab->copie){
                pab→detruire(pnb→data);
        }
        free(pnb);
        }
}
```

```
void detruire_ab(PArbreBinaire pab) {
  detruire_noeud_binaire(pab->racine, pab);
  free(pab);
}
```

question 6:

```
void afficher_ab_prefixe_rec(PNoeudBinaire pnb, PArbreBinaire pab) {
	if (pnb) {
		printf("( ");
		pab->afficher(pnb->data);
		afficher_ab_prefixe_rec(pnb->gauche,pab);
		afficher_ab_prefixe_rec(pnb->droit,pab);
		printf(") ");
	}
}


void afficher_ab_prefixe(PArbreBinaire pab) {
	afficher_ab_prefixe_rec(pab→racine, pab);
}
```

question 7 : décaler l’appel d’affichage

Exercice 2 :

question 1:

```
PNoeudBinaire ajouter_abr_rec(PNoeudBinaire pnb, PArbreBinaire pab,void
*data) {
	if(pnb) {
	if (pab->comparer(pnb->data,data)>0){
		pnb→gauche=ajouter_abr_rec(pnb→gauche,pab,data);
	}
	else {
		if (pab→comparer(pnb→data,data)<0)
		{
		pnb→droit=ajouter_abr_rec(pnb→droit,pab,data);
		}
	return pnb;
	}
	else {
		return creer_noeud_binaire(pab, data);
		}
	}
}

void ajouter_abr(PArbreBinaire pab,void *data) {
	pab→racine=ajouter_abr_rec(pab→racine,  pab, data);
}
```

question 2 :

```
PNoeudBinaire chercher_abr_rec(PNoeudBinaire pnb, PArbreBinaire pab,
void *data) {
	if(pnb) {
		if (pab->comparer(pnb->data,data)>0)
			return chercher_abr_rec(pnb→gauche,pab,data);
		else if (pab->comparer(pnb->data,data)<0)
			return chercher_abr_rec(pnb->droit,pab,data);
		else
		return pnb;
	}
	else
	return NULL;
}

PNoeudBinaire chercher_abr(PArbreBinaire pab, void *data) {
	return chercher_abr_rec(pab->racine,pab,data);
}
```

question 3 :

```
void lire_abr(PArbreBinaire pab, const char *nom_fichier) {
	FILE *f=fopen(nom_fichier,"r");
	if (f==NULL) {
	printf("Probleme lors de l’ouverture du fichier.\n");
	return;
	}
	pab->copie=0;
	void *data;
	while((data=pab->lire(f))) {
		ajouter_abr(pab,data);
	}
	fclose(f);
	pab->copie=1;
// pour que la memoire allouee pendant la lecture soit liberee correctement a
la fin...
}
```

question 4 :

```
void ecrire_ab_rec(PNoeudBinaire pnb, PArbreBinaire pab, FILE *f) {
        if (pnb) {
        pab->ecrire(pnb->data,f);
        fprintf(f,"\n");
        ecrire_ab_rec(pnb->gauche,pab,f);
        ecrire_ab_rec(pnb->droit,pab,f);
        }
}


void ecrire_ab(PArbreBinaire pab, const char *nom_fichier) {
FILE *f=fopen(nom_fichier,"w");
if (f==NULL) {

void ecrire_ab(PArbreBinaire pab, const char *nom_fichier) {
        FILE *f=fopen(nom_fichier,"w");
        if (f==NULL) {
        printf("Probleme lors de l’ouverture du fichier.\n");
        return;
        }
        ecrire_ab_rec(pab->racine,pab,f);
        }
        fclose(f);
```