```c
#include <stdlib.h>
#include <assert.h>
#include "arn.h"

ARN arn_creer(void) {
    ARN t = malloc(sizeof(struct ARN_t));
    t->racine = NULL;
    return t;
}

void arn_inserer(ARN t, void* valeur, compare_fct compare) {
    Noeud noeudAInserer = nouveauNoeud(valeur, ROUGE, NULL, NULL);
    if (t->racine == NULL) {
        t->racine = noeudAInserer;
    } else {
        Noeud n = t->racine;
        while (1) {
            int comparaisonResultat = compare(valeur, n->valeur);
            if (comparaisonResultat == 0) {
                n->valeur = valeur;
                free (noeudAInserer);
                return;
            } else if (comparaisonResultat < 0) {
                if (n->gauche == NULL) {
                    n->gauche = noeudAInserer;
                    break;
                } else {
                    n = n->gauche;
                }
            } else {
                assert (comparaisonResultat > 0);
                if (n->droite == NULL) {
                    n->droite = noeudAInserer;
                    break;
                } else {
                    n = n->droite;
                }
            }
        }
        noeudAInserer->parent = n;
    }
    insertionCas1(t, noeudAInserer);
}

void arn_supprimer(ARN t, void* valeur, compare_fct compare) {
    Noeud fils;
    Noeud n = chercherNoeud(t, valeur, compare);
    if (n == NULL) return; // on n'a pas trouvé la valeur, on fait rien

    if (n->gauche != NULL && n->droite != NULL) {
        Noeud pred = noeudMaximum(n->gauche);
        n->valeur = pred->valeur;
        n = pred;
    }

    assert(n->gauche == NULL || n->droite == NULL);
    fils = n->droite == NULL ? n->gauche  : n->droite;
    if (couleurNoeud(n) == NOIR) {
        n->couleur = couleurNoeud(fils);
        suppressionCas1(t, n);
    }
    remplacerNoeud(t, n, fils);
    if (n->parent == NULL && fils != NULL)
        fils->couleur = NOIR;

    free(n);
}

void arn_afficherOrdreCroissant(ARN t, afficher_fct afficher) {
    afficherNoeudsOrdreCroissant(t->racine, afficher);
}
```

```
    void arn_ecrireFichierDot(ARN t, stringAffichage_fct afficherNoeud, FILE* fichierDot) {
        fprintf(fichierDot, "digraph G {\n");
75      ecrireNoeudsFichierDot(t->racine, afficherNoeud, fichierDot);
        fprintf(fichierDot, "\n}\n");
    }
```