# Einfache Listen: kopf kopf,ende

|  |  |
| --- | --- |
| void durchlaufen(struct knoten\* kopf) {  knoten\* laufzeiger = kopf;  while (laufzeiger != NULL) {  printf("%d",laufzeiger->wert);  laufzeiger = laufzeiger->next;  }  return 0;  } | void durchlaufen(struct knoten \*kopf) {  struct knoten \*laufzeiger;  laufzeiger = kopf;  while (laufzeiger != NULL) {  printf("%d ", laufzeiger->wert);  laufzeiger = laufzeiger->next;  }  } |
| struct knoten\* suchen(struct knoten\* kopf, int gesuchter\_wert) {  knoten\* laufzeiger = kopf;  if ( kopf == NULL) return -1;    while (laufzeiger != NULL && laufzeiger->wert != gesuchter\_wert) {  laufzeiger = laufzeiger->next;  }  return laufzeiger;  } | struct knoten \*suchen(struct knoten \*kopf, int gesuchter\_wert){  struct knoten \*laufzeiger;  laufzeiger = kopf;  while ((laufzeiger != NULL) && (laufzeiger->wert != gesuchter\_wert)){  laufzeiger = laufzeiger->next;  }  return laufzeiger;  } |
| /\* Liefert -1 im Fehlerfall, sonst 0 \*/  int einfuegen\_kopf(struct knoten\* \*kopfref, struct knoten\* einzufueg) {  // if (kopfref == NULL) kopfref = einzufueg; //eig unoetig  if (einzufueg == NULL ) return -1;  einzufueg->next = \*kopfref;  \*kopfref = einzufueg;  return 0;  } | /\* Liefert -1 im Fehlerfall, sonst 0 \*/  int einfuegen\_kopf(struct knoten \*\*kopfref, struct knoten \*\*enderef, struct knoten \*einzufueg) {  if ((einzufueg == NULL) || (kopfref == NULL) || (enderef == NULL))  return -1;  einzufueg->next = \*kopfref;  \*kopfref = einzufueg;  return 0;  } |
| int einfuegen\_nach(struct knoten\* nachdiesem, struct knoten\* einzufueg) {  if(nachdiesem == NULL || einzufueg == NULL) return -1;  einzufueg->next = nachdiesem->next;  nachdiesem->next = einzufueg;  return 0;  } | int einfuegen\_nach(struct knoten \*nachdiesem, struct knoten \*einzufueg, struct knoten \*\*enderef) {  if ((nachdiesem == NULL) || (einzufueg == NULL) || (enderef == NULL))  return -1;  einzufueg->next = nachdiesem->next;  nachdiesem->next = einzufueg;  // ---  if (einzufueg->next == NULL) {  \*enderef = einzufueg;  }  // ---  return 0;  } |
| int einfuegen\_ende(struct knoten\* \*kopfref, struct knoten\* einzufueg) {  if ( kopfref == NULL || einzufueg == NULL) return -1;  if ( \*kopfref == NULL) {  \*kopfref = einzufueg;  einzufueg->next = NULL;  } else {  knoten\* laufzeiger = \*kopfref;  while(laufzeiger->next != NULL) {  laufzeiger = laufzeiger->next;  }  laufzeiger->next = einzufueg;  einzufueg->next = NULL;  }  } | int einfuegen\_ende(struct knoten \*\*kopfref, struct knoten \*\*enderef, struct knoten \*einzufueg) { //O(1)  if ((einzufueg == NULL) || (kopfref == NULL) || (enderef == NULL)|| (\*enderef == NULL))  return -1;  if (\*kopfref == NULL) {  einzufueg->next = NULL;  \*kopfref = einzufueg;  \*enderef = einzufueg;  } else {  (\*enderef)->next = einzufueg;  einzufueg->next = NULL;  \*enderef = einzufueg;  }  return 0;  } |

|  |  |
| --- | --- |
| /\* Liefert den entfernten Knoten oder NULL \*/  struct knoten\* entfernen\_kopf(struct knoten\* \*kopfref) {  if ((kopfref==NULL)||(\*kopfref==NULL)) return NULL;  knoten\* temp = \*kopfref;  temp = temp->next;  \*kopfref = temp;  return kopfref;  } | /\* Liefert den entfernten Knoten oder NULL \*/  struct knoten \*entfernen\_kopf(struct knoten \*\*kopfref, struct knoten \*\*enderef) {  struct knoten \*kopf\_alt;  if ((kopfref == NULL) || (\*kopfref == NULL) || (enderef == NULL))  return NULL;  kopf\_alt = \*kopfref;  \*kopfref = (\*kopfref)->next;  return kopf\_alt;  } |
| struct knoten\* entfernen\_ende(struct knoten\* \*kopfref) {    if ((kopfref==NULL)||(\*kopfref==NULL)) return NULL;  knoten\* laufzeiger = \*kopfref;  if(laufzeiger->next == NULL){  \*kopfref = NULL;  return laufzeiger;  }  while ( (laufzeiger->next)->next != NULL){  laufzeiger = laufzeiger->next;  }  laufzeiger->next == NULL;  return laufzeiger;  } | struct knoten \*entfernen\_ende(struct knoten \*\*kopfref, struct knoten \*\*enderef) {  struct knoten \*vor\_ende, \*ende;  if ((kopfref == NULL) || (\*kopfref == NULL) || (enderef == NULL))  return NULL;  if ((\*kopfref)->next == NULL) {  ende = \*kopfref;  \*kopfref = NULL;  return ende;  }  vor\_ende = \*kopfref;  while (vor\_ende->next->next != NULL) {  vor\_ende = vor\_ende->next;  }  ende = vor\_ende->next;  vor\_ende->next = NULL;  // ---  \*enderef = vor\_ende;  // --  return ende;  } |
| struct knoten\* entfernen(struct knoten\* \*kopfref, struct knoten\* auszufueg) {  knoten\* laufzeiger = \*kopfref;  if(\*kopfref == NULL || kopfref == NULL || auszufueg == NULL) return -1;  if (auszufueg == \*kopfref) {  \*kopfref = (\*kopfref)->next;  return auszufueg;  }  while ((laufzeiger->next) != auszufueg ) {  if (laufzeiger->next == NULL) return NULL;  laufzeiger = laufzeiger->next;  }  laufzeiger->next = auszufueg->next;  return auszufueg;  } | struct knoten \*entfernen(struct knoten \*\*kopfref, struct knoten \*\*enderef, struct knoten \*auszufueg) {  struct knoten \*vor\_auszufueg;  if ((auszufueg == NULL) || (kopfref == NULL) || (\*kopfref == NULL) || (enderef == NULL))  return NULL;  if (auszufueg == \*kopfref) {  \*kopfref = (\*kopfref)->next;  return auszufueg;  }  vor\_auszufueg = \*kopfref;  while (vor\_auszufueg->next != auszufueg) {  if (vor\_auszufueg->next == NULL)  return NULL;  vor\_auszufueg = vor\_auszufueg->next;  }  vor\_auszufueg->next = auszufueg->next;  // ---  if (vor\_auszufueg->next == NULL) {  \*enderef = vor\_auszufueg;  }  // ---  return auszufueg;  } |