```c
/* AvlTest.c */
#include <stdio.h>
#include <string.h>
#include <malloc.h>
#include <stdlib.h>
#include "Avl.h"

static int IntCmp(const void *keyAddr, const void *dataAddr)
{
    int *p1 = (int *)keyAddr;
    int *p2 = (int *)dataAddr;
    return (*p1 - *p2);
}

static void IntTraverse(void *dataAddr)
{
    int *p = (int *)dataAddr;
    printf("%d\n", *p);
}

static int StringCmp(const void *keyAddr, const void *dataAddr)
{
    char *p1 = *(char **)keyAddr;
    char *p2 = *(char **)dataAddr;
    return strcmp(p1, p2);
}

static void StringTraverse(void *dataAddr)
{
    char *p = *(char **)dataAddr;
    printf("%s\n", p);
}

static void StringFree(void *dataAddr)
{
    char *p = *(char **)dataAddr;
    free(p);
}

int main()
{
    AVLTREE intAvl;
    AvlNew(&intAvl, sizeof(int), IntCmp, NULL);

    int i = 0;
    for (; i < 10; i++)
    {
        AvlInsert(&intAvl, &i);
    }
    if (!AvlEmpty(&intAvl))
    {
        printf("intAvl size is %d\n", AvlSize(&intAvl));
        printf("intAvl height is %d\n", AvlHeight(&intAvl));
        printf("intAvl traverse pre:\n");
        AvlTravPre(&intAvl, IntTraverse);
        printf("intAvl traverse in:\n");
        AvlTravIn(&intAvl, IntTraverse);
        printf("intAvl traverse post:\n");
        AvlTravPost(&intAvl, IntTraverse);
        printf("intAvl traverse level:\n");
        AvlTravLevel(&intAvl, IntTraverse);
    }
    int intRemove = 1;
    if (0 == AvlRemove(&intAvl, &intRemove))
    {
        printf("intAvl remove key %d success\n", intRemove);
    }
    else
    {
        printf("intAvl remove key %d fail\n", intRemove);
    }
    if (0 == AvlRemove(&intAvl, &intRemove))
    {
        printf("intAvl remove key %d success\n", intRemove);
    }
    else
    {
        printf("intAvl remove key %d fail\n", intRemove);
    }
    if (!AvlEmpty(&intAvl))
    {
        printf("intAvl size is %d\n", AvlSize(&intAvl));
        printf("intAvl height is %d\n", AvlHeight(&intAvl));
        AvlTravInRec(&intAvl, IntTraverse);
    }

    int intSearch = 2;
    AVLNODE *node = AvlSearch(&intAvl, &intSearch);
    if (NULL != node)
    {
        printf("key %d is in intAvl\n", intSearch);
    }
    else
    {
        printf("key %d is not in intAvl\n", intSearch);
    }
    intSearch = 11;
    node = AvlSearch(&intAvl, &intSearch);
    if (NULL != node)
    {
        printf("key %d is in intAvl\n", intSearch);
    }
    else
    {
        printf("key %d is not in intAvl\n", intSearch);
    }
    AvlDispose(&intAvl);

    printf("\n\n");

    AVLTREE stringAvl;
    AvlNew(&stringAvl, sizeof(char *), StringCmp, StringFree);
    char *name1 = strdup("pc");
    char *name2 = strdup("pcwl513");
    char *name3 = strdup("pcpc");
    char *name4 = strdup("jerry");
    char *name5 = strdup("jerry.peng");
    char *name6 = strdup("yanglupu");
    char *name7 = strdup("zhanglei");
    char *name8 = strdup("lishanke");
    AvlInsert(&stringAvl, &name1);
    AvlInsert(&stringAvl, &name2);
    AvlInsert(&stringAvl, &name3);
    AvlInsert(&stringAvl, &name4);
    AvlInsert(&stringAvl, &name5);
    AvlInsert(&stringAvl, &name6);
    AvlInsert(&stringAvl, &name7);
    AvlInsert(&stringAvl, &name8);
    if (!AvlEmpty(&stringAvl))
    {
        printf("stringAvl size is %d\n", AvlSize(&stringAvl));
        printf("stringAvl height is %d\n", AvlHeight(&stringAvl));
        printf("stringAvl traverse pre:\n");
        AvlTravPre(&stringAvl, StringTraverse);
        printf("stringAvl traverse in:\n");
        AvlTravIn(&stringAvl, StringTraverse);
        printf("stringAvl traverse post:\n");
        AvlTravPost(&stringAvl, StringTraverse);
        printf("stringAvl traverse level:\n");
        AvlTravLevel(&stringAvl, StringTraverse);
    }

    char *strRemove = "pcpc";
    if (0 == AvlRemove(&stringAvl, &strRemove))
    {
        printf("stringAvl remove key %s success\n", strRemove);
```

```c
    }
    else
    {
        printf("stringAvl remove key %s fail\n", strRemove);
    }
    if (0 == AvlRemove(&stringAvl, &strRemove))
    {
        printf("stringAvl remove key %s success\n", strRemove);
    }
    else
    {
        printf("stringAvl remove key %s fail\n", strRemove);
    }
    if (!AvlEmpty(&stringAvl))
    {
        printf("stringAvl size is %d\n", AvlSize(&stringAvl));
        printf("stringAvl height is %d\n", AvlHeight(&stringAvl));
        AvlTravInRec(&stringAvl, StringTraverse);
    }

    char *strSearch = "yanglupu";
    node = AvlSearch(&stringAvl, &strSearch);
    if (NULL != node)
    {
        printf("key %s is in stringAvl\n", strSearch);
    }
    else
    {
        printf("key %s is not in stringAvl\n", strSearch);
    }
    strSearch = "123";
    node = AvlSearch(&stringAvl, &strSearch);
    if (NULL != node)
    {
        printf("key %s is in stringAvl\n", strSearch);
    }
    else
    {
        printf("key %s is not in stringAvl\n", strSearch);
    }
    AvlDispose(&stringAvl);
    return 0;
}
```