DAA ASSIGNMENT

```
#include <stdio.h>

int max(int a, int b) {
    return (a > b) ? a : b;
}

int knapsack(int W, int wt[], int val[], int n) {
    int dp[n+1][W+1];

    for (int i = 0; i <= n; i++) {
        for (int w = 0; w <= W; w++) {

            if (i == 0 || w == 0)
                dp[i][w] = 0;

            else if (wt[i-1] > w)
                dp[i][w] = dp[i-1][w];

            else
                dp[i][w] = max(dp[i-1][w],
                               val[i-1] + dp[i-1][w - wt[i-1]]);
        }
    }
    return dp[n][W];
}
```

```
int main() {
    int n, W;

    printf("Enter number of items: ");
    scanf("%d", &n);

    int wt[n], val[n];

    printf("Enter weights of items:\n");
    for (int i = 0; i < n; i++)
        scanf("%d", &wt[i]);

    printf("Enter values of items:\n");
    for (int i = 0; i < n; i++)
        scanf("%d", &val[i]);

    printf("Enter maximum capacity of knapsack: ");
    scanf("%d", &W);

    int result = knapsack(W, wt, val, n);

    printf("Maximum value in knapsack = %d\n", result);

    return 0;
}
```

C daaassignment.c ●

C daaassignment.c > knapsack(int, int [], int [], int)

```c
#include <stdio.h>

int max(int a, int b) {
    return (a > b) ? a : b;
}

int knapsack(int W, int wt[], int val[], int n) {
    int dp[n+1][W+1];

    for (int i = 0; i <= n; i++) {
        for (int w = 0; w <= W; w++) {

            if (i == 0 || w == 0)
                dp[i][w] = 0;

            else if (wt[i-1] > w)
                dp[i][w] = dp[i-1][w];

            else
                dp[i][w] = max(dp[i-1][w],
                               val[i-1] + dp[i-1][w - wt[i-1]]);
        }
    }
    return dp[n][W];
}

int main() {
    int n, W;

    printf("Enter number of items: ");
    scanf("%d", &n);

    int wt[n], val[n];
```

daaassignment.c ●

daaassignment.c > knapsack(int, int [], int [], int)

```c
#include <stdio.h>

int max(int a, int b) {
    return (a > b) ? a : b;
}

int knapsack(int W, int wt[], int val[], int n) {
    int dp[n+1][W+1];

    for (int i = 0; i <= n; i++) {
        for (int w = 0; w <= W; w++) {

            if (i == 0 || w == 0)
                dp[i][w] = 0;

            else if (wt[i-1] > w)
                dp[i][w] = dp[i-1][w];

            else
                dp[i][w] = max(dp[i-1][w],
                               val[i-1] + dp[i-1][w - wt[i-1]]);
        }
    }
    return dp[n][W];
}

int main() {
    int n, W;

    printf("Enter number of items: ");
    scanf("%d", &n);

    int wt[n], val[n];
```

```
PS C:\Users\afifa\OneDrive\Desktop\random> cd "c:\Users\afifa\OneDrive\Desktop\random\" ; if ($?) { gcc da
aassignment.c -o daaassignment } ; if ($?) { .\daaassignment }
c:/mingw/bin/../lib/gcc/mingw32/6.3.0/../../../libmingw32.a(main.o):(.te›
nce to `WinMain@16'
collect2.exe: error: ld returned 1 exit status
PS C:\Users\afifa\OneDrive\Desktop\random> cd "c:\Users\afifa\OneDrive\De
aassignment.c -o daaassignment } ; if ($?) { .\daaassignment }
Enter number of items: 5
Enter weights of items:
50 60 70 80 90
Enter values of items:
100 200 225 222 111
Enter maximum capacity of knapsack: 150
Maximum value in knapsack = 447
PS C:\Users\afifa\OneDrive\Desktop\random>
```

Snipping Tool

Screenshot copied to clipboard