实验四：

一．

1. 编写程序，将两个升序的序列合并成另一个升序的序列。
2. 思路分析：1.赋给一个数组，将数组排序。
3. 按顺序各判断两序列的大小，将较小的赋给新数列，并 将赋值过的序列号加1。
4. 代码实现：

#include<stdio.h>

#define NUMBER 5

void exp(){

int a[NUMBER] = {1,3,5,7,9};

int b[NUMBER] = {2,4,6,8,10};

int c[NUMBER \* 2];

int i = 0,j = 0,k = 0;

while( i<NUMBER && j<NUMBER ){

if( a[i] < b[j] )

c[k++] = a[i++];

else

c[k++] = b[j++];

}

while( i < NUMBER ) c[k++] = a[i++];

while( j < NUMBER ) c[k++] = b[j++];

while( i = 0 ; i < 2 \* NUMBER ;i++)

printf("%d\n");

}

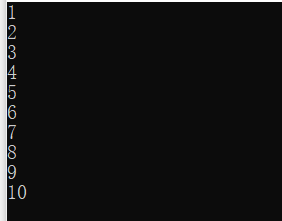
int main(){

exp();

return 0;

}

1. 运行截图：



二．

1. 背包问题：给定n个重量为{w1,w1,...,wn},价值为{v1,v2,...,vn}的物品和一个最大承重为W的背包，求解能够放入背包中且获得价值最大的方案。
2. 思路分析：暴力解法，遍历所有不超过最大承重的放法（即遍历所有子集）计算最大价值。（可用二进制代表子集中是否含有该元素）
3. 代码实现：

#include<stdio.h>

#include<math.h>

#include<string.h>

#define NUM 5

#define MAXWEIGHT 10

void exp1(){

int item[2][NUM] = {{6,3,5,4,6},{2,2,6,5,4}};

int subset[NUM] = {0};

int tmp[NUM];

int i,j,k,s;

int upper = (int)pow(2,NUM);

int sumvalue = 0,maxvalue = 0,sumweight = 0;

for( i = 1 ; i < upper ; i++ ){

//每进行一次循环，将总重和总价和集合置0

s = i;

sumvalue = 0 ;

sumweight = 0;

memset(subset,0,NUM \* sizeof(int));

for( j = 0 ; s != 0 ; j++ ){

subset[j] = s % 2;

s = (unsigned int ) s / 2;

}

for( k = 0 ; k < NUM ; k++ ){

if(subset[k] == 1){

sumweight += item[1][k];

if(sumweight <= MAXWEIGHT ) sumvalue += item[0][k];

else break;

}

}

if( k == NUM && maxvalue < sumvalue){

maxvalue = sumvalue;

memcpy(tmp,subset,NUM \* sizeof(int));

}

}

for(i = 0 ; i < NUM ; i++ ){

if(tmp[i] == 1)

printf("%d\n", i+1 );

}

printf("%d",maxvalue);

}

int main(){

exp1();

return 0;

}

1. 运行截图：

’