

## CP 111 Computer Programming, 2/2554, Section B01/B02

### Programming Homework 07 Solution

#### 1. Problem name: Sort3\_PassByRef

```
#include <stdio.h>
void Sort3(int *px1, int *px2, int *px3);

int main()
{
    int a,b,c;

    scanf("%d %d %d",&a,&b,&c);
    Sort3(&a,&b,&c);
    printf("%d %d %d\n",a,b,c);
    return 0;
}

void Sort3(int *px1, int *px2, int *px3)
{
    int tmp;

    if(*px1 > *px2)
    {
        tmp=*px1;
        *px1=*px2;
        *px2=tmp;
    }
    if(*px2 > *px3)
    {
        tmp=*px2;
        *px2=*px3;
        *px3=tmp;
    }
    if(*px1 > *px2)
    {
        tmp=*px1;
        *px1=*px2;
        *px2=tmp;
    }
}
```

## 2. Problem name: Baht\_Change

```
#include <stdio.h>
```

```
void Baht_Change(int money, int *B1000, int *B500, int *B100,  
                 int *B50, int *B20, int *C10, int *C5, int *C2, int *C1);
```

```
int main()
```

```
{
```

```
int i_money, B1000, B500, B100, B50, B20, C10, C5, C2, C1;
```

```
    scanf("%d", &i_money);
```

```
    Baht_Change(i_money, &B1000, &B500, &B100, &B50, &B20, &C10, &C5, &C2, &C1) ;
```

```
    printf("%d\n", B1000);
```

```
    printf("%d\n", B500);
```

```
    printf("%d\n", B100);
```

```
    printf("%d\n", B50);
```

```
    printf("%d\n", B20);
```

```
    printf("%d\n", C10);
```

```
    printf("%d\n", C5);
```

```
    printf("%d\n", C2);
```

```
    printf("%d\n", C1);
```

```
    return 0;
```

```
}
```

```
void Baht_Change(int money, int *B1000, int *B500, int *B100,  
                 int *B50, int *B20, int *C10, int *C5, int *C2, int *C1)
```

```
{
```

```
    *B1000=money/1000;
```

```
    money=money-(*B1000)*1000;
```

```
    *B500=money/500;
```

```
    money=money-(*B500)*500;
```

```
    *B100=money/100;
```

```
    money=money-(*B100)*100;
```

```
    *B50=money/50;
```

```
    money=money-(*B50)*50;
```

```
    *B20=money/20;
```

```
    money=money-(*B20)*20;
```

```
    *C10=money/10;
```

```
    money=money-(*C10)*10;
```

```
    *C5=money/5;
```

```
    money=money-(*C5)*5;
```

```
    *C2=money/2;
```

```
    *C1=money-(*C2)*2;
```

```
}
```

### 3. Problem name: Pos\_Neg\_Avg\_Std\_Func

```
#include <stdio.h>
#include <math.h>
int Pos_Neg_Avg_Std_Func(int *, int, double *, double *,double *, double *);
int main()
{
    int N,x;
    int data[100];
    double pos_avg,pos_std,neg_avg,neg_std;
    N=0;
    do
    {
        scanf("%d",&x);
        if(x!=0)
        {
            data[N]=x;
            N=N+1;
        }
    }
    while(x!=0);
    Pos_Neg_Avg_Std_Func(data,N,&pos_avg,&pos_std,&neg_avg,&neg_std);
    printf("%.2f %.2f\n",pos_avg,pos_std);
    printf("%.2f %.2f\n",neg_avg,neg_std);
    return 0;
}
int Pos_Neg_Avg_Std_Func(int *x, int N, double *pos_avg, double *pos_std,
                        double *neg_avg, double *neg_std)
{
    int pos_sum,neg_sum,N_pos,N_neg,i;
    double pos_sum_sq;
    double neg_sum_sq;
    pos_sum=0;
    neg_sum=0;
    N_pos=0;
    N_neg=0;
    for(i=0;i<N;i++)
    {
        if(x[i]>0){
            pos_sum=pos_sum+x[i];
            N_pos++;
        }
        else
        {
            neg_sum=neg_sum+x[i];
            N_neg++;
        }
    }
    *pos_avg=(double)pos_sum/N_pos;
    *neg_avg=(double)neg_sum/N_neg;
    pos_sum_sq=0.0;
    neg_sum_sq=0.0;
    for(i=0;i<N;i++)
    {
        if(x[i]>0)
            pos_sum_sq=pos_sum_sq+((x[i]-*pos_avg)*(x[i]-*pos_avg));
        else
            neg_sum_sq=neg_sum_sq+((x[i]-*neg_avg)*(x[i]-*neg_avg));
    }
    *pos_std=sqrt(pos_sum_sq/N_pos);
    *neg_std=sqrt(neg_sum_sq/N_neg);
}
```

#### 4. Problem name: Reverse\_String

```
#include <stdio.h>
#define MAX_STR_LEN 500

void ReverseStr(char *instr, char *outstr);

int main()
{
    char str1[MAX_STR_LEN+1];
    char str2[MAX_STR_LEN+1];

    gets(str1);

    ReverseStr(str1, str2);

    printf("%s\n", str2);

    return 0;
}

void ReverseStr(char *instr, char *outstr)
{
    int i, k, last;

    i=0;
    while(instr[i]!=0)
        i++;
    last=i-1;

    k=0;
    for(i=last; i>=0; i--)
    {
        ostr[k]=instr[i];
        k++;
    }
    ostr[k]=0;
}
```

## 5. Problem name: Insert\_string

```
#include <stdio.h>
void Insert_String(char *instr, char *Ins, int pos, char *outstr);
#define MAX_STR_LEN 501
int main()
{
    char str1[MAX_STR_LEN];
    char inserted[MAX_STR_LEN];
    char str2[MAX_STR_LEN];
    int pos;

    gets(str1);
    gets(inserted);
    scanf("%d", &pos);

    Insert_String(str1, inserted, pos, str2);

    printf("%s\n", str2);

    return 0;
}

void Insert_String(char *instr, char *Ins, int pos, char *outstr)
{
    int i, k;
    // copy instr[0 to pos-1] -> ostr
    k=0;
    for(i=0; i<pos; i++)
    {
        ostr[k]=instr[i];
        k++;
    }

    // concat Ins -> ostr
    i=0;
    while(Ins[i]!=0)
    {
        ostr[k]=Ins[i];
        k++;
        i++;
    }

    // concat instr[pos to END OF String] -> ostr
    i=pos;
    while(instr[i]!=0)
    {
        ostr[k]=instr[i];
        k++;
        i++;
    }

    // Append the NULL character to ostr
    ostr[k]=0;
}
```

## 6. Problem name: Twenty\_One

```
#include <stdio.h>
void Sort3(int *px1, int *px2, int *px3);
int main()
{
    int i,j,k,X[10];
    int X1,X2,X3,maxsum,sum;

    for(i=0;i<10;i++){
        scanf("%d",&X[i]);
    }
    maxsum=0;
    for(i=0;i<10;i++)
    {
        for(j=0;j<10;j++)
        {
            for(k=0;k<10;k++)
            {
                if(i!=j && j!=k && i!=k)
                {
                    sum=X[i]+X[j]+X[k];
                    if(sum>maxsum && sum<=21)
                    {
                        maxsum=sum;
                        X1=X[i];
                        X2=X[j];
                        X3=X[k];
                    }
                }
            }
        }
    }
    Sort3(&X1,&X2,&X3);
    printf("%d %d %d %d\n",X1,X2,X3,maxsum);
    return 0;
}

void Sort3(int *px1, int *px2, int *px3)
{
    int tmp;

    if(*px1 > *px2)
    {
        tmp=*px1;
        *px1=*px2;
        *px2=tmp;
    }
    if(*px2 > *px3)
    {
        tmp=*px2;
        *px2=*px3;
        *px3=tmp;
    }
    if(*px1 > *px2)
    {
        tmp=*px1;
        *px1=*px2;
        *px2=tmp;
    }
}
```

## 7. Problem name: Min\_Max\_in\_Range

```
#include <stdio.h>
#include <math.h>
#define MAX_DATA 100

int main()
{
    int X[MAX_DATA];
    int i, N, sum, min, max;
    double avg, sum2, std;

    scanf("%d", &N);
    for(i=0; i<N; i++)
        scanf("%d", &X[i]);
    sum=0;
    for(i=0; i<N; i++)
    {
        sum=sum+X[i];
    }
    avg=(double)sum/N;
    sum2=0;
    for(i=0; i<N; i++)
    {
        sum2=sum2+(X[i]-avg)*(X[i]-avg);
    }
    std=sqrt(sum2/N);

    printf("%.2f\n%.2f\n", avg, std);

    min=(int)(avg+2*std);
    max=(int)(avg-2*std);
    for (i=0; i<N; i++)
    {
        if(X[i]>=(avg-std) && X[i]<=(avg+std))
        {
            if(X[i]<min)
                min=X[i];
            if(X[i]>max)
                max=X[i];
        }
    }

    printf("%d\n%d\n", min, max);
    return 0;
}
```

### 8. Problem name: TrimStar

```
#include <stdio.h>
#define MAX_STR_LEN 500

void TrimStar(char *instr1, char *instr2);

int main()
{
    char str1[MAX_STR_LEN+1];
    char str2[MAX_STR_LEN+1];

    gets(str1);
    TrimStar(str1, str2);
    printf("%s\n", str2);
    return 0;
}

void TrimStar(char *instr, char *outstr)
{
    int i, k, first, last;

    i=0;
    while(instr[i]!='*')
    {
        i++;
    }
    first=i;

    i=0;
    while(instr[i]!=0)
        i++;
    i=i-1;
    while(instr[i]!='*')
    {
        i--;
    }
    last=i;

    k=0;
    for(i=first; i<=last; i++)
    {
        ostr[k]=instr[i];
        k++;
    }
    ostr[k]=0;
}
```