C Programming Language Exam 2015/9/15(120 minutes)

1. (A)

2. It's possible to insert a node anywhere in a linked list, and remove a node from anywhere in it. However, nodes in a stack may only be inserted at the top of the stack and remove from the top of a stack.

3. A queue has pointers to both its head and its tail so that nodes may be inserted at the tail and deleted from the head. A stack has a single pointer to the top of the stack where both insertion and deletion of nodes is perfomed.

4. Non-recursive :

```
#include<stdio.h>
int main()
{
    int big,small,temp,ans;
    scanf("%d %d",&big,&small);
    do{
        if(small>big)
        {
            temp=big;
            big=small;
            small=big;
        }
        ans=big%small;
        big=small;
        small=ans;
    }while(small!=0);
    printf("%d",big);
    return 0;
}
```

Recursive :

```
#include<stdio.h>
int gcd(int big,int small)
{
       int temp=0;
       temp=big%small;
       if(temp==0)
              return small;
       else
              gcd(small,temp);
}
int main()
{
      int a,b,temp,ans;
      scanf("%d %d",&a,&b);
      if(b>a)
      {
            temp=a;
            a=b;
            b=temp;
      }
      ans=gcd(a,b);
      printf("%d",ans);

      return 0;
}
```

5.

a)

```
#include <stdio.h>
int main()
{
   char num1[22];
   char num2[19];

   int i_num2[18];
int ans[21];
int i=0;
for(i=0;i<21;i++)
{
   ans[i]=0;
}

for(i=0;i<18;i++)
{
   i_num2[i]=0;
}
printf("input the first number\n");
scanf("%21s",num1);

printf("input the second number\n");
scanf("%18s",num2);

for(i=0;i<21;i++)
{
   ans[i]=(int)num1[i]-48;
}

for(i=0;i<18;i++)
{
   i_num2[i]=(int)num2[i]-48;
}

for(i=17;i>=0;i--)
{
   ans[i+3]=ans[i+3]+i_num2[i];
   if(ans[i+3]>9)
   {
      ans[i+3]=ans[i+3]-10;
      ans[i+2]++;
   }
}

for(i=0;i<21;i++)
```

```
   printf("%d ",ans[i]);

   return 0;
}
```

b)

```
#include <stdio.h>
int main()
{
   char num1[22];
   char num2[9];

   int i_num2[8];
   int ans[21];
   int i=0;
   for(i=0;i<21;i++)
   {
      ans[i]=0;
   }

   for(i=0;i<8;i++)
   {
      i_num2[i]=0;
   }
   printf("input the first number\n");
   scanf("%21s",num1);

   printf("input the second number\n");
   scanf("%8s",num2);

   for(i=0;i<21;i++)
   {
      ans[i]=(int)num1[i]-48;
   }

   for(i=0;i<8;i++)
   {
      i_num2[i]=(int)num2[i]-48;
```

```
    }

    for(i=7;i>=0;i--)
    {
       if(ans[i+13]<i_num2[i])
       {
          ans[i+13]=ans[i+13]+10;
          ans[i+12]--;
       }
       ans[i+13]=ans[i+13]-i_num2[i];
    }

    for(i=0;i<21;i++)
    printf("%d ",ans[i]);

    return 0;
 }
```

6.
```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   int i=0;
   char letter[100];

while(i<100&&(letter[99-i]=getchar())&&letter[99-i]!=EOF)
{
   if(letter[99-i]=='\n')
   {
         i-=1;
         for(;i>=0;i--)
         printf("%c",letter[99-i]);
         printf("\n");
         i=0;
         continue;
   }
```

```
 if(letter[99-i]<'A'||(letter[99-i]>'Z'&&letter[99-i]<'a')||letter[99
 -i]>'z')
{
          i++;
          continue;
}
     letter[99-i]-=5;
     if(letter[99-i]<'A')
               letter[99-i]+=26;
     else if(letter[99-i]>'Z'&&letter[99-i]<'a')
               letter[99-i]+=26;
     else;
     i++;
}
     return 0;
}
```

7.
```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
int solution[6]={0};
bool judge()
{
     int check[7]={0},i;
     for(i=0;i<6;i++)
     check[solution[i]]++;
     for(i=1;i<7;i++)
     if(check[i]>1)
          return false;
     return true;
}
void fill(int place)
{
     if(place==6&&solution[0]+solution[1]+solution[2]==sol
     ution[2]+solution[3]+solution[4]&&solution[2]+solution
     [3]+solution[4]==solution[4]+solution[5]+solution[0])
```

```
    printf("   %d   \n %d %d \n%d %d
    %d\n\n",solution[0],solution[1],solution[5],solution[2],s
    olution[3],solution[4]);
    int i,j;
    for(i=1;i<=6;i++)
    {
        solution[place]=i;
        if(judge()==true&&place<6)
            fill(place+1);
        solution[place]=0;
    }
}
int main()
{
    fill(0);
    return 0;
}
```

8.
```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int
    special_award_1,special_award_2,award_1,award_2,a
    ward_3,more_award_1,more_award_2,more_award_3,n
    umber;
    printf("輸入特別獎號碼: ");
    scanf("%d",&special_award_1);
    printf("輸入特獎號碼: ");
    scanf("%d",&special_award_2);
    printf("輸入頭獎號碼: ");
    scanf("%d %d %d",&award_1,&award_2,&award_3);
    printf("輸入增開六獎號碼: ");
```

```
scanf("%d %d
%d",&more_award_1,&more_award_2,&more_award_3
);
printf("輸入您的發票號碼: ");
while(scanf("%d",&number)!=EOF)
{
    if(number==special_award_1)
            printf("恭喜中特別獎 1000 萬元\n");
    else if(number==special_award_2)
            printf("恭喜中特獎 200 萬元\n");
    else if
    (number==award_1||number==award_2||number==a
    ward_3)
            printf("恭喜中頭獎 20 萬元\n");
    else if
    (number%10000000==award_1%10000000||number
    %10000000==award_2%10000000||number%100000
    00==award_3%10000000)
            printf("恭喜中二獎 4 萬元\n");
else if
(number%1000000==award_1%1000000||number%1000
000==award_2%1000000||number%1000000==award_3
%1000000)
            printf("恭喜中三獎 1 萬元\n");
else if
(number%100000==award_1%100000||number%100000
==award_2%100000||number%100000==award_3%1000
00)
            printf("恭喜中四獎 4 千元\n");
```

```
        else if
        (number%10000==award_1%10000||number%10000==award_2%10000||number%10000==award_3%10000)
                printf("恭喜中五獎 1 千元\n");
        else if
        (number%1000==award_1%1000||number%1000==award_2%1000||number%1000==award_3%1000)
                printf("恭喜中六獎 2 百元\n");
        else if
        (number%1000==more_award_1||number%1000==more_award_2||number%1000==more_award_3)
                printf("恭喜中增開六獎 2 百元\n");
        else
            printf("槓龜\n");

        printf("\n 輸入您的發票號碼: ");
        }
        return 0;
    }
```

9. a)

```
        GradeNodePtr startptr = NULL;
```

b)

```
        GradeNodePtr newPtr;
       newPtr = malloc( sizeof( GradeNode ) );
       strcpy( newPtr->lastName, ”Jones” );
       newPtr->grade = 91.5;
       newPtr->nextPtr = NULL;
```

c)

To insert ”Adams”:
previousPtr is NULL, currentPtr points to the first element in the list.

```
newPtr->nextPtr = currentPtr;
```

```
startPtr = newPtr;
To insert “Thompson”:
previousPtr points to the last element in the list
(containing ”Smith”)
currentPtr is NULL.
newPtr->nextPtr = currentPtr;
previousPtr->nextPtr = newPtr;
To insert ”Pritchard”:
previousPtr points to the node containing “Jones”
currentPtr points to the nodes containing “Smith”
newPtr->nextPtr = currentPtr;
previousPtr->nextPtr = newPtr;
```

d)

```
currentPtr = startPtr;
while( currentPtr != NULL )
{
     Printf( “Lastname = %s\nGrade = %6.2f\n”,
     currentPtr->lastname, currentPtr->grade );
     currentPtr = currentPtr->nextPtr;
}
```

e)

```
 currentPtr = startPtr;
while( currentPtr != NULL )
{
     tempPtr = currentPtr;
     currentPtr = currentPtr->nextPtr;
     free( tempPtr );
 }
 startPtr = NULL;
```

10. (1)Pointer p points to one memory block,while q points to another.After q is assigned to p,both variables now point to the second memory block.

There are no pointers to the first block,so we’ll never be able to use it again. We call the program has a memory leak.

```
p = malloc(...);
q = malloc(...);
p = q;
```

(2)Use free function to release unneeded memory.

```
p = malloc(...);
q = malloc(...);
```

11.
```
#include <stdio.h>
int main(void)
{
    int number;
    int i,j;

    scanf("%d",&number);
    int S[number];
    for(i=0;i<number;i++)
    {
    scanf("%d",&S[i]);
    }

    int total=(1+number)*number/2;
    int possibleMax[total];

    int count=0,tmp;
    for(i=0;i<number;i++)
    {
    tmp=S[i];
    possibleMax[count++]=tmp;
    for(j=i+1;j<number;j++)
    {
        tmp=tmp*S[j];
        possibleMax[count++]=tmp;
    }
    }

    int max=0;
    for(i=0;i<total;i++)
    if(possibleMax[i]>max)
    max=possibleMax[i];

    printf("%d\n",max);

    return 0;
}
```

12.

```
#include <stdio.h>
#include <stdbool.h>
#define N 25
int calc(int,int,int*,int*,int*,int*);
int main(void)
{
    char firstLine[N+1]={0};
    int i,j,k;
    for(i=0;i<N+1;i++)
    {
        scanf("%c",&firstLine[i]);
        if(firstLine[i]=='\n')
            break;
    }

    int length=i;
    char matrix[length][length];
    for(i=0;i<length;i++)
    matrix[0][i]=firstLine[i];

    for(i=1;i<length;i++)
    for(j=0;j<length;j++)
    {
        scanf("%c",&matrix[i][j]);
    if(matrix[i][j]=='\n')
        j--;
}

int count0=0,count1=0;
for(i=0;i<length;i++)
for(j=0;j<length;j++)
{
    if(matrix[i][j]=='1')
        count1++;
    else
        count0++;
}

if(count0==0 && count1>0)
{
    printf("%d\n",length*length);
    return 0;
}
else if(count0>0 && count1==0)
{
    printf("0\n");
    return 0;
}

int zerosRow[count0];
int zerosCol[count0];
k=0;
for(i=0;i<length;i++)
{
for(j=0;j<length;j++)
{
    if(matrix[i][j]!='1')
    {
        zerosRow[k]=i;
        zerosCol[k]=j;
        k++;
    }
    }

    int largestArea[count1];
    k=0;
    for(i=0;i<length;i++)
        for(j=0;j<length;j++)
            if(matrix[i][j]=='1')
            {
                largestArea[k++]=
                calc(i,j,&length,
                &count0,zerosRow,zerosCol);
            }

    int max=0;
    for(i=0;i<count1;i++)
        if(largestArea[i]>max)
            max=largestArea[i];

    printf("%d\n",max);

    return 0;
}

int calc(int row,int col,
int* matrixLength,
int* count0,
int* zerosRow,
int* zerosCol)
{
    int i,j,k;
    bool check[*matrixLength - row][*matrixLength -col];
    for(i=row;i<*matrixLength;i++)
        for(j=col;j<*matrixLength;j++)
        {
            check[i-row][j-col]=true;
            for(k=0;k<*count0;k++)
            {
                if(zerosRow[k]<row || zerosCol[k] <col)
                continue;
                if(i>=zerosRow[k] && j>=zerosCol[k])
                {
                    check[i-row][j-col]=false;
                    break;
                }
            }
        }

    int max=0,tmp;
    for(i=0;i< *matrixLength - row;i++)
        for(j=0;j< *matrixLength -col;j++)
            if(check[i][j]==true)
            {
                tmp=(i+1)*(j+1);
                if(tmp>max)
                    max=tmp;
            }

    return max;
}
```