

进制转换

```
1  #include <iostream>
2  #include <stack>
3  using namespace std;
4
5  int main(){
6      stack<int> s;
7      int n;
8      scanf("%d",&n);
9      while(n!=0){
10         s.push(n%2);
11         n=n/2;
12     }
13     while(s.empty()!=1){
14         cout<<s.top();
15         s.pop();
16     }
17
18 }
```

1353：表达式括号匹配(stack)

```
1  #include <iostream>
2  #include <string>
3  #include <stack>
4  //括号匹配
5  using namespace std;
6  stack <int> s;
7  int main(){
8      string v;
9      bool flag=false;
10     cin>>v;
11     for (int i=0;i<=v.length();i++){
12         char t=v[i];
13         if (t=='@')
14             break;
15         if (t=='(')
16             s.push(1);
17         else if (t==')'&&s.size())
18             s.pop();
19         else if (t==')'&&!s.size())
20             flag=true;
21     }
22     if (s.size())
23         flag=true;
24     // cout<<flag;
25     string ans=flag?"NO":"YES";
26     cout<<ans<<endl;
27     return 0;
28 }
```

1354：括弧匹配检验

```
1  #include<iostream>
2  #include<cstdio>
3  #include<cstring>
4  #include<stack>
5  //括号匹配
6  using namespace std;
7  char str[10000];
8  stack<int> S; //用1代表(, 2代表), 3代表[, 4代表]
9  int main(){
10     cin>>str;
11     int len=strlen(str);
12     for(int i=0;i<len;i++)
13     {
14         if(str[i]=='(') //记录左圆括号
15             S.push(1);
16         else if(str[i]==')') //记录右圆括号
17         {
18             if(S.empty()) //栈为空
19                 S.push(2);
20             else if(S.top()==1)
21                 S.pop();
22             else
23                 S.push(2);
24         }
25         else if(str[i]=='[') //记录左方括号
26             S.push(3);
27         else if(str[i]==']') //记录右方括号
28         {
29             if(S.empty()) //栈为空
30                 S.push(4);
31             else if(S.top()==3)
32                 S.pop();
33             else
34                 S.push(4);
35         }
36     }
37     if(S.empty())
38         printf("OK");
39     else
40         printf("Wrong");
41     return 0;
42 }
```

1331：【例1-2】 后缀表达式的值

```
1  //后缀表达式求值
2  #include <iostream>
3  #include <cstring>
4  using namespace std;
5  int stack[101];
6  char s[256];
7  int compute(char s[256]){
```

```

8     int i=0,top=0,x;
9     while(i<=strlen(s)-2){
10         switch (s[i]){
11             case '+':stack[--top]+=stack[top+1];break;//int
temp=stack[top+1]+stack[top];top--;stack[top]=temp;
12             case '-':stack[--top]-=stack[top+1];break;
13             case '*':stack[--top]*=stack[top+1];break;
14             case '/':stack[--top]/=stack[top+1];break;
15             default:
16                 x=0;
17                 while(s[i]!=' '){
18                     x=x*10+s[i++]-'0';
19                 }
20                 stack[++top]=x;
21                 break;
22             }
23             i++;
24         }
25         return stack[top];
26     }
27 int main(){
28     cin.getline(s,256);
29     cout<<compute(s);
30     return 0;
31 }
32

```

1369：合并果子(fruit)

```

1  #include <iostream>
2  #include <queue>
3  #include <cstdio>
4  using namespace std;
5  int n;
6  priority_queue<int,vector<int>,greater<int> > h;//升序排列，小顶堆
7  //priority_queue<int, vector<int>, less<int> > a;降序排列，大顶堆
8  void work(){
9      int i,x,y,ans=0;
10     cin>>n;
11     for(i=1;i<=n;i++){
12         cin>>x;
13         h.push(x);
14     }
15     for(i=1;i<n;i++){
16         x=h.top();h.pop();
17         y=h.top();h.pop();
18         ans+=x+y;
19         h.push(x+y);
20     }
21     cout<<ans<<endl;
22 }
23
24 int main(){
25     work();

```

```
26     return 0;
27 }
28
```

单调队列

队列：先进先出

优先队列：优先级高的元素进队时插队

单调队列：队列始终满足从队首开始，优先级递减，进队时间越靠后越晚；新元素进队时，将优先级低于自己的直接踢出队。

应用：动态规划优化

1597：【例 1】滑动窗口

核心思想：高效地排除不可能成为答案的数字。

后来的数字大于前面的数字，则前面的数字不可能再成为答案。

每个数字进队时踢出前面比自己大的数，始终取队首为答案。

单开一个数组记录队里数在原数组的下标，队首数字太老就出队。

P1886 滑动窗口

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  const int maxx=1000001;
4  long long int N,K,data[maxx],dl[maxx],id[maxx],le=1,ri;
5  int main(){
6      scanf("%lld%lld",&N,&K);
7      for(long long int i=1;i<=N;i++)
8          scanf("%lld",&data[i]);
9      for(long long int i=1;i<=N;i++){
10         while(le<=ri&&data[i]<dl[ri])
11             ri--;
12         ri++;
13         dl[ri]=data[i];
14         id[ri]=i;
15         if(id[le]+K<=i)
16             le++;
17         if(i>=K)
18             printf("%lld ",dl[le]);
19     }
20     printf("\n");
21     memset(dl,0x3f3f3f3f,sizeof(dl));
22     le=1,ri=0;
23     for(long long int i=1;i<=N;i++){
24         while(le<=ri&&data[i]>dl[ri])
25             ri--;
26         ri++;
27         dl[ri]=data[i];
28         id[ri]=i;
29         if(id[le]+K<=i)
30             le++;
```

```
31         if(i>=K)
32             printf("%lld ",dl[le]);
33     }
34     printf("\n");
35 }
```