

```

Cutrod(p,n){
  if(n==0) return 0;
  q = -1000000;
  for (i=1;i<=n;i++){
    q = max(q, p[i] + cutrod(p,n-1));
  }
  return q;
}

```

recursive

```

cutrod(p,n){
  r[1,...,n] = {0};
  return MemCutRodAux(p,n);
}


```

top down

```

MemCutRodAux(p,n){
  if (n==0) return 0;  r[0] = 0;
  if(r[n]>0) return r[n];
  else{
    q = -100000000;
    for(i=1;i<=n;i++){
      q = max(q,p[i] + MemCutRodAux(p, n-i));
    }
    r[n] = q;
    return q;
  }
}

```



Lab 8: Dynamic Programming

Q1 Write the recursive function, top-down dynamic programming, and bottom-up dynamic programming to calculate the maximum value for the rod cutting problem.

```

int cr_recursive ( int [] p, int n);
int cr_top_down_dp (int [] p, int n);
int cr_bottom_up_dp ( int [] p, int n);

```

where p is the price list and n is the length of the rod. For example, if the prices of different lengths are given in the following table, and the length of the rod is 9, the maximum revenue will be 25.

Length	1	2	3	4	5	6	7	8	9	10
Price SGD	1	5	8	9	10	17	17	20	24	30

Q2 Modify the bottom up dynamic programming in Q2 to print the list of lengths of cutting pieces (in ascending order) achieved in the optimal solution and return the maximum revenue.

```

int cr_bottom_up_dp_print ( int [] p, int n);

```

For example, with the example in Q2, the rod will be cut into two pieces with the lengths 3 and 6, and the maximum revenue is 25.

Q3 Consider a game where a player can score 3, 5, or 10 points in one move. Given a total score N, you need to write a function to find the total number of unique ways to reach a score of N. Please solve the problem in linear time.

```

int waysToScore (int n);

```

Q3

```

int memo[100];

int main(){
  for(i=0;i<100;i++){
    memo[i] = -1;
  }
  int countways(int n){
    {
      if(n==0){
        return 1;
      }
      if(memo[n]!=-1){
        return memo[n];
      }
    }
    int ways = countways(n-3)+ countways(n-5) + countways(n-10);
    memo[n] = ways;
    return ways;
  }
}

```

bottom up

```

DPCutRod(p,n){
  r[1,...,n] = {0};
  for(j=1;j<=n;j++){
    for(i=1;i<=j;i++){
      r[j] = max(r[j], p[i] + r[j-i]);
    }
  }
  return r[n];
}

```

8-1

Q2

```

int cr_bottom_up_dp_print(int *p,int n)
{
  int i,j;
  r[0]=0;

  int firstCut[n+1];
  for(i=0;i<=n;i++){
    int position = 0;
    for(j=0;j<=i;j++){
      int temp = p[j] +r[i-j];
      if(temp > r[i]){
        position = j;
        r[i] = temp;
        firstCut[i] = position;
      }
    }
  }

  printf("Length of each piece is:
  \n");
  printpos(n,firstCut);
  printf("\n");
  return r[n];
}

```