

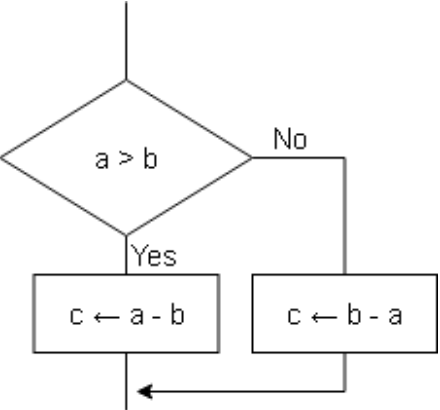
問題 1. 実数変数 a、b、c が宣言されている。a=5.0, b=2.0, c=1.0 で初期化されている時、以下の値を求めなさい。

- (1) a と c の和
- (2) a を b で割った剰余
- (3) a と b の積
- (4) a,b,c の合計
- (5) a,b,c の平均
- (6) a / b

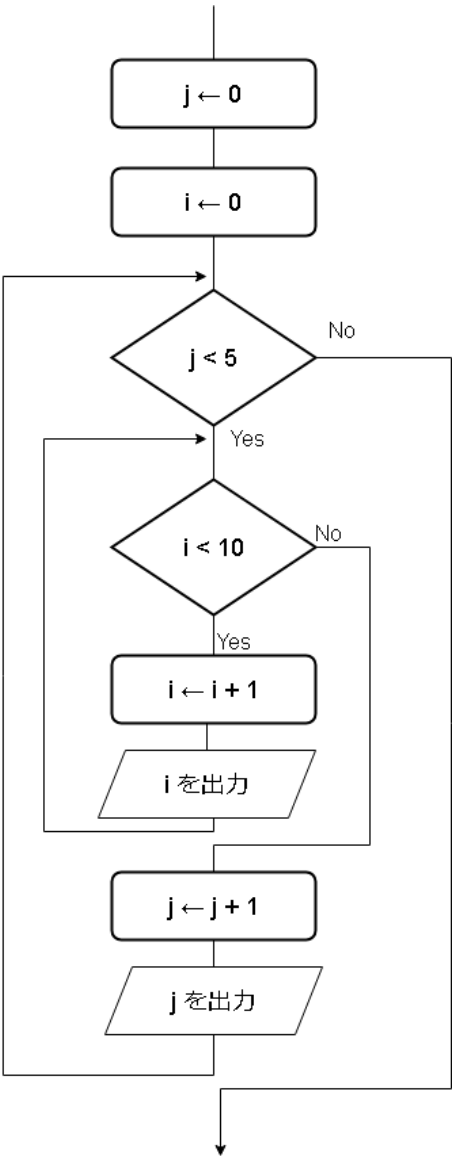
問題 2. プログラムを構成する3つの基本構造を答えなさい。

問題 3. 次のフローチャートを疑似言語で書きなさい。

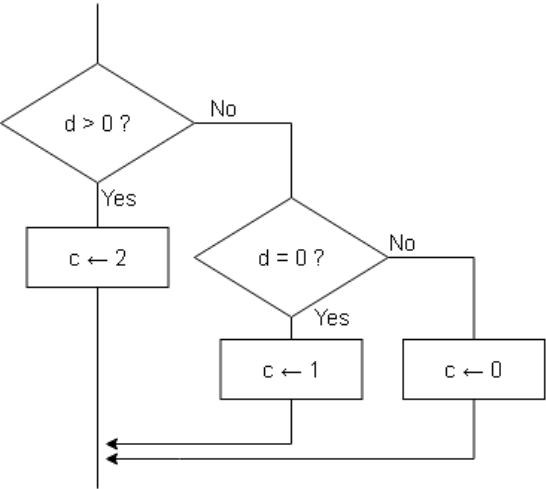
(1)



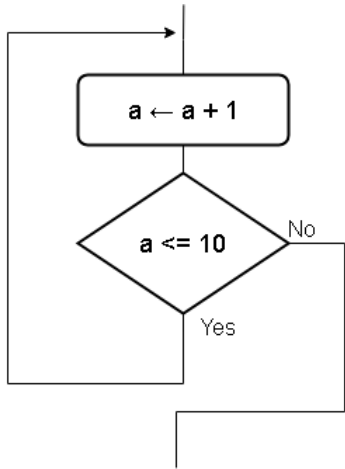
(2)



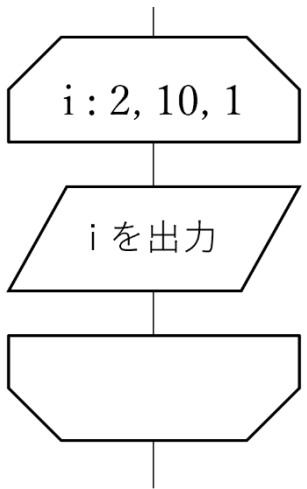
(3)



(4)



(5)



〔宣言、注釈及び処理〕

記述形式		説明
○		手続、変数などの名前、型などを宣言する。
/* 文 */		文に注釈を記述する。
処 理	・変数 ← 式	変数に式の値を代入する。
	・手続(引数, …)	手続を呼び出し、引数を受け渡す。
	▲ 条件式 処理	単岐選択処理を示す。 条件式が真のときは処理を実行する。
	▲ 条件式 処理 1 ↓ 処理 2	双岐選択処理を示す。 条件式が真のときは処理 1 を実行し、偽のときは処理 2 を実行する。
	■ 条件式 処理	前判定繰返し処理を示す。 条件式が真の間、処理を繰り返し実行する。
	■ 処理 ■ 条件式	後判定繰返し処理を示す。 処理を実行し、条件式が真の間、処理を繰り返し実行する。
■ 変数: 初期値, 条件式, 増分 処理		繰返し処理を示す。 開始時点で変数に初期値 (式で与えられる) が格納され、条件式が真の間、処理を繰り返す。また、繰り返すごとに、変数に増分 (式で与えられる) を加える。

問題 4．整列アルゴリズムについて以下の問に答えなさい。

アルゴリズムの疑似コード		
<p>○Sort①(A[], N)</p> <p>○int N: 配列の要素数</p> <p>○int A: 0～(N-1)の配列</p> <p>Sort①(A, N)</p> <pre>■ for i ← 1 to N-1 •v ← A[i] •j ← i - 1 ■ j >= 0 && A[j] > v •A[j+1] ← A[j] •j-- ■ •A[j+1] = v ■</pre>	<p>○Sort②(A[], N)</p> <p>○void swap(X, Y):配列中の X と Y の要素を交換する操作</p> <p>○int N: 配列の要素数</p> <p>○int A: 0～(N-1)の配列</p> <p>Sort②(A[N])</p> <pre>■ for i ← 0 to N-1 •minj ← i ■ for j ← i to N-1 ▲ if A[j] < A[minj] •minj ← j ▼ ■ •swap(A[i], A[minj]) ■</pre>	<p>○Sort③(A[], N)</p> <p>○void swap(X, Y):配列中の X と Y の要素を交換する操作</p> <p>○int N: 配列の要素数</p> <p>○int A:</p> <p>0～(N-1)の要素を持つ配列</p> <p>○bool flag:</p> <p>繰り返し制御のためのフラグ</p> <p>Sort③(A[], N)</p> <pre>flag ← true ■ flag = false •flag ← false ■ for j ← N-1 to 1 ▲ A[j] < A[j-1] •swap(A[j],A[j-1]) •flag ← true ▼ ■ ■</pre>
実行結果		
<p>結果①</p> <p>6 1 7 5 2 3 4 8</p> <p>1 6 7 5 2 3 4 8</p> <p>1 2 7 5 6 3 4 8</p> <p>1 2 3 5 6 7 4 8</p> <p>1 2 3 4 6 7 5 8</p> <p>1 2 3 4 5 7 6 8</p> <p>1 2 3 4 5 6 7 8</p>	<p>結果②</p> <p>6 1 7 5 2 3 4 8</p> <p>1 6 7 5 2 3 4 8</p> <p>1 6 7 5 2 3 4 8</p> <p>1 5 6 7 2 3 4 8</p> <p>1 2 5 6 7 3 4 8</p> <p>1 2 3 5 6 7 4 8</p> <p>1 2 3 4 5 6 7 8</p> <p>1 2 3 4 5 6 7 8</p>	<p>結果③</p> <p>6 1 7 5 2 3 4 8</p> <p>1 6 2 7 5 3 4 8</p> <p>1 2 6 3 7 5 4 8</p> <p>1 2 3 6 4 7 5 8</p> <p>1 2 3 4 6 5 7 8</p> <p>1 2 3 4 5 6 7 8</p>
整列アルゴリズム名		
(あ) 挿入ソート	(い) マージソート	(う) 選択ソート
(え) クイックソート	(お) シェルソート	(か) バブルソート
英語表記		
(a) shell sort	(b) merge sort	(c) bubble sort
(d) selection sort	(e) insertion sort	(f) quick sort

問1．表を見て、疑似コード、結果、アルゴリズム名、とその英語表記の正しい組み合わせを選び記号で解答用紙に答えなさい

問2．以下の文章は表の整列アルゴリズム名のいずれかの説明を書いたものである、アルゴリズム名（あ）～（か）に対応する正しいものを記号で答えなさい。（ダミーもあるので注意）

- ① 既に整列済みのデータ列の正しい位置に、データを追加する操作を繰り返していく方法である。
- ② データ中の最小値を求め、次にそれを除いた部分の中から最小値を求める。この操作を繰り返していく方法である。
- ③ 適当な基準値を選び、それより小さい値のグループと大きい値のグループにデータを分割する。同様に、グループの中から基準値を選び、それぞれのグループを分割する。この操作を繰り返していく方法である。
- ④ 未整列の部分を順序木に構成し、そこから最大値又は最小値を取り出して既整列の部分に移す操作を繰り返して、未整列部分を縮めていく方法
- ⑤ 隣り合ったデータの比較と入替えを繰り返すことによって、小さな値のデータを次第に端の方に移していく方法である。
- ⑥ データ列を 2 分割し、2 分割したデータ列に対し整列を行い、分割したデータ列を合わせて(マージして) 整列を行う方法
- ⑦ 一定間隔おきに取り出した要素から成る部分列をそれぞれ整列し、更に間隔を詰めて同様の操作を行い、間隔が 1 になるまでこれを繰り返す方法

問3．整列アルゴリズムの計算量のオーダが $O(n^2)$ のものを3つ、 $O(n \log(n))$ のものを2つ「整列アルゴリズム名」から選び記号で答えなさい。

問題 5．線系探索のアルゴリズムは以下の疑似言語で表すことができる。

- 検索対象のデータ列： A(n)
- 検索するデータ： Key
- データ列の数（配列数）： N
- 戻り値：
 - ・データが見つかった時：データの位置(index)
 - ・データが見つからない時：-1 (Not Foundのしるしとしてあり得ないindexを返す)

SequentialSearch()

```
• i ← 0
■ i < N and A[i] != Key
|   • i ← i + 1
■
▲ i => N
|   • return -1
+-----+
|   • return i
▼
```

番兵法は、配列の末尾に検索キー (key) を配置しておくことで、確実にループを終わらせることができ、1 回あたりの比較処理も減らすことができるという改良法である。

具体的には、

- データ列を要素数 N より 1 つ多く用意し、末尾に key を代入

(index は 0 ~ N)

• key が index=N で発見されたときは 元のデータ列には key が存在しなかった時=番兵に引っかけた時である

• key が N より小さい index で見つかった時 (key = A[index]) は、key が発見できた時なのでその時の index を返す

上の疑似コードを、番兵法を使ったものに書き換えなさい。

問題 6. 次のような構造体を使ったプログラムを作りたい。RPG ゲームのキャラクタを表している (つもり) である。ソースコードを読んで問いに答えなさい。

```
//キャラクタのHP (生命力): 整数
//キャラクタのMP (魔法力): 整数
//キャラクタの攻撃力: 実数
//キャラクタの防御力: 実数

struct mychar{
    int hp, mp; //HP と MP
    float atk, def; //攻撃力と防御力
};
```

問 1. mychar 型の変数、hero と mob を宣言する宣言文を書きなさい。

問 2. mychar 型の引数 _mc で受け取ったキャラクタの各パラメータを表示する関数 void printStatus を以下のように宣言した。(A) には同じものが記述されている。(A) の内容として適切なものを選びなさい。

```
void printStatus(mychar _mc)
{
    cout << " HP: " << (A)_____hp << endl; //HP の表示
    cout << " MP: " << (A)_____mp << endl; //MP の表示
    cout << " ATK: " << (A)_____atk << endl; //攻撃力の表示
    cout << " DEF: " << (A)_____def << endl; //防御力の表示
}
```

(ア) hero. (イ) mob. (ウ) _mc. (エ) _mc-> (オ) *(_mc).

問 3. mychar 型の引数 _mc で受け取った構造体に値をセットする関数 setStatus を以下のように宣言した。(A) には同じものが記述されている。(A) の内容として適切なものをすべて選びなさい。

```
void setStatus(mychar *_mc, int _hp, int _mp, float _atk, float _def)
{
    (A)_____hp = _hp; //HP をセット
    (A)_____mp = _mp; //MP をセット
    (A)_____atk = _atk; //ATK をセット
    (A)_____def = _def; //DEF をセット
}
```

(ア) (*_mc). (イ) *(mob) (ウ) _mc. (エ) _mc-> (オ) *(_mc).

問 4. 問 1 で宣言した mychar 型の 2 つの変数 hero に以下のようなパラメータをセットしたい。ここまでで宣言した適切な関数を使って値をセットしなさい。

```
hero.hp: 100
hero.mp: 50
hero.atk: 80
hero.def: 50
```

問 5. 問 3 の宣言では setStatus 関数に mychar 型の引数を渡すときにアドレス渡しを行っている。これは、いくつかの理由から値渡し (通常の渡し方) ではなく、アドレス渡しにしている。以下の文章を読み理由として適切なものをすべて選びなさい。

- (ア) 実引数を値で渡すと、関数内で渡された構造体を破壊する恐れがあるため。
- (イ) 構造体の実引数を値で渡すと、関数内で構造体のコピーが生成され、その構造体に値を代入しても関数を抜けるときに構造体が破棄されるため。
- (ウ) 引数の値渡しによる構造体のコピーは、構造体が大きくなるとメモリの読み書きが増え処理が遅くなったり、メモリを圧迫したりするため推奨されない。
- (エ) 引数にポインタを指定して、構造体のアドレスを渡すと内部で構造体を再定義できるため、様々な処理ができて自由度が高くなる。
- (オ) 実引数のアドレス渡しをすることにより、構造体の配列を扱うことができるため複数のキャラクタを自在に操ることができる。
- (カ) 実引数をアドレスで渡すことにより、そのアドレスの先にある値をコピーすることなく直接書き換えることができる。そのため、引数で指定した構造体をポインタ経由で変更するめにアドレス渡しを行う。
- (キ) 構造体を関数に渡すときは、アドレス渡しにしないと複数の引数を扱うことができないため。
- (ク) C++ の関数では戻り値は、1 つの値しかリターンできない、戻り値の代用として、ポインタで複数の引数を渡すことにより、渡された複数の値を関数内で変更できる。

問題 7. 次の文はコンピュータにまつわる様々なデータ構造や用語について述べた文章である。何について述べている文章か内容から判断し、下の語群から適切な言葉を選び記号で答えなさい

- (ア) 連続信号を離散化し、符号化の過程を経て 2 値の信号へ変換すること。
- (イ) プログラミング言語のうち、ソースコードを一括して機械語へ変換する方式のプログラミング言語の分類のこと
- (ウ) ソースコードを一括して機械語へ変換するプログラムの事
- (エ) 符号はズラさずに 2 進数の値を左右へ何ビットかズラす操作の事
- (オ) 2 進数の値を $0 \rightarrow 1$ 、 $1 \rightarrow 0$ のようにすべてのビットを逆の値にする操作
- (カ) メモリ中に同じ型の領域を連続でとることにより、同じ型のデータを複数並べ、先頭アドレスからの何個目のデータ化を示す index を使ってそれぞれの要素にアクセスできる (ランダムアクセスが可能)
- (キ) コンピュータ上で、様々な数値を扱うための内部形式の規格の事、整数であれば内部では 2 進数表現、負の数は 2 の補数表現、実数は IEEE754 浮動小数点形式であらわす、など表現するデータにより、内部形式が決まっている。
- (ク) プログラムを作るうえでの値を格納する入れ物の事
- (ケ) 1 つのデータに複数の項目を持つデータ構造の事。例として、学生一人のデータに対し、学籍番号、名前、性別、などを持つ形式
- (コ) レコード型を C++ のプログラムで表現したいときに使うデータ形式
- (サ) データとデータを、1 方向につなげて、順番にアクセスするデータ構造。データの追加や削除が高速に行えるが、ランダムアクセスができない。C++ では構造体とポインタで実現する。
- (シ) データとデータを双方向につなげて、自分の前か後のデータをたどっていくことで、目的のデータへアクセスするデータ構造。追加や削除が高速に行えるが、ランダムアクセスはできない。C++ では構造体とポインタで実現する。
- (ス) 1 次元配列で 1 列に格納されたデータを、格納した順番に取り出すデータ構造。「先入れ先出し方式」「FIFO (First In First Out)」と言われる。
- (セ) 1 次元配列で格納されたデータを、最後に入れたデータから先に取り出すデータ構造。常に一番新しいデータから順番に使う構造になっており、「後入れ先出し方式」「LIFO (Last In First Out)」方式であると言われる。
- (ソ) データ (ノード) とデータをエッジで接続したデータ構造。エッジに方向性を持ったものと、持っていないものがある。
- (タ) 根からデータが枝によって接続されている形のデータ構造。
- (チ) 木構造のうち、ある 1 つの親の子が高々 2 つしかない。というルールで形成されたものの
- (ツ) 木構造のすべてのデータを漏らさずにたどること。
- (テ) グラフや木の巡回法のうち、葉に到達したら親に戻り、親の別の子を葉に到達するまでたどり、を繰り返すことで達成する手法
- (ト) グラフや木の巡回法のうち、根に近いほうから順に深さを増しながらたどっていくことで達成する手法

語群：

- | | |
|------------------|---------------|
| (1) 幅優先探索 | (16) 木構造 |
| (2) 配列 | (17) 二分探索 |
| (3) キュー | (18) 構造体 |
| (4) 二分木 | (19) コンパイラ |
| (5) 巡回 (トラバース) | (20) 補数 |
| (6) 深さ優先探索 | (21) ビット反転 |
| (7) スタック | (22) コンパイラ型言語 |
| (8) グラフ | (23) ランダムシード |
| (9) 単方向リスト | (24) 算術シフト |
| (10) 論理シフト | (25) 自己参照構造体 |
| (11) プッシュ | (26) 量子化 |
| (12) 整数 | (27) 型 |
| (13) A/D 変換 | (28) レコード型 |
| (14) 変数 | (29) 双方向リスト |
| (15) ビットフラグ | (30) インタープリタ |

問題 8. 次の逆ポーランド記法で書かれた数式の値を中置記法に変換しなさい。中置記法で書かれたものは逆ポーランド記法に変換しなさい。

- (1) 1 2 +
- (2) 3 4 * 2 5 * -
- (3) 3 + (2 * 4)
- (4) a c d + * a c + /
- (5) 3 4 + 7 / 2 6 + 4 / + 3 /

問題 9. 最近のゲームやタブレットや PC に関するニュースなどを見て、興味のあるアルゴリズムや技術について書きなさい (得点になります)。また、ゲームアルゴリズムの授業について、要望や感想などあれば書きなさい (今後の授業の参考にはしますが、良くも悪くも点数にはなりません)

これで問題は終わりです。お疲れ様 (^_^)