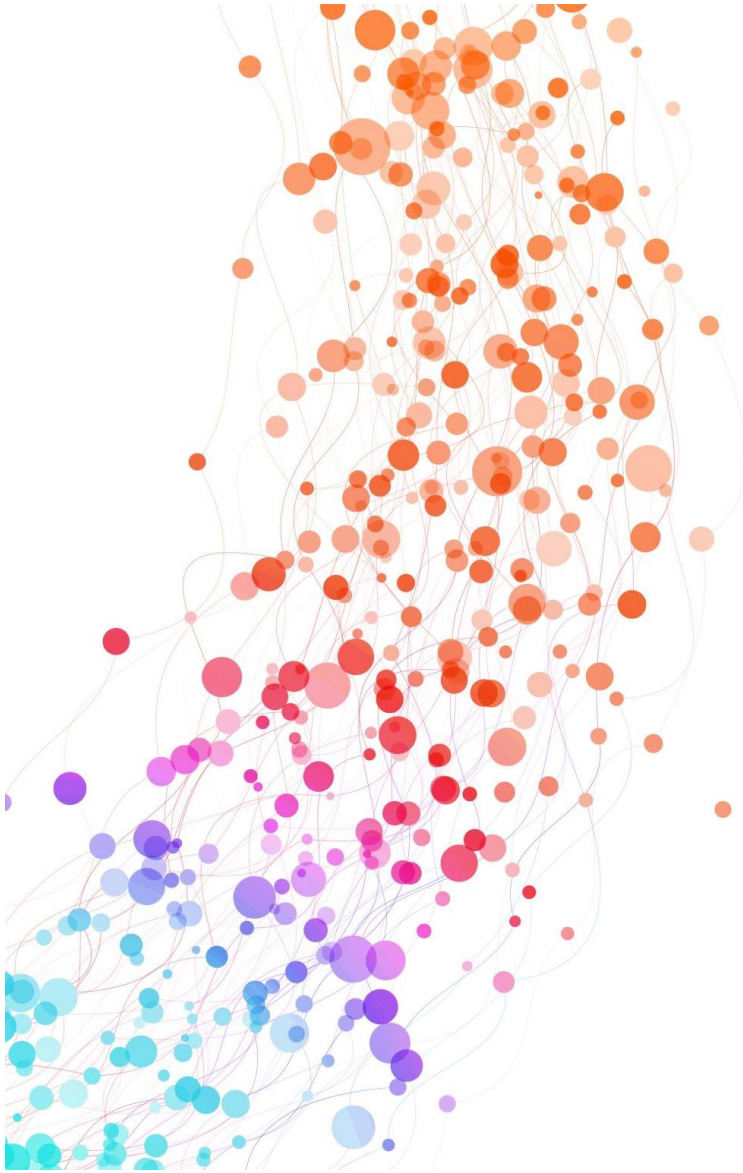
駿台_2024
パック問題

3年情報



アセンブラ言語と高水準言語について

●コンピューターが理解しやすい言語→（① アセンブリ言語　）
　低水準言語や機械語とも呼ばれる

●人間が理解しやすい言語→（②　高水準言語　　）と呼ばれる

pythonなどはこちら

高水準言語の種類について

- pythonなどのプログラムはプログラムを実行した後、コンピュータが理解しやすいように機械語に翻訳される

（① インタプリタ言語　）・・・処理をひとつずつ翻訳しながら実行

例）処置①→処理①が終われば処理②

●代表的な言語・・・（JavaScript、Python）

（② コンパイラ言語　）・・・全部まとめて翻訳してから実行

例）処置①、処理②をまとめて実行

●代表的な言語・・・（C、C++、Java）

オブジェクト指向について

（① オブジェクト指向　　）・・・

コンピュータの処理をパーツに分けて「オブジェクト」と定義し、
これらのオブジェクトを組み合わせてシステム全体を構築する考え

●代表的な言語・・・（JavaScript、Python、Java、C++、C#）

欠点　各パーツに分けているので、各手順ごとのような処理別に
　　対処するとなると難しい

●プログラム解説

問1

問1　次の生徒（S）と先生（T）の会話文を読み，空欄［ア］に当てはまる数字をマークせよ。また，空欄［イ］に入れるのに最も適当なものを，後の解答群のうちから一つ選べ。なお，「%」は整数の除算における余りを求める演算子であり，「==」は「等しい」ことを表す比較演算子である。

S：ある自然数 n が素数かどうかを判定するにはどうすればいいか考えてみたいと思います。

T：ではまず n が素数であることの定義から確認しましょう。

S：ええと，n が2以上の自然数で，正の約数が1と n しかもたないことです。

T：そうですね。ではまず n に数値を代入して，それが素数かどうかを判定するプログラムを作成してみてください。n を2から順に割っていって余りが0となることがあれば素数ではなく，最後まで0となることがなければ素数であると判定するようなプログラムがよいでしょう。

S：図1のようなプログラムを作成してみました。(09) 行目はループを強制終了するためのコードです。また変数は以下のものを用いました。

> 変数 **sosuhantei**…素数かどうかを判定する変数。1のときは素数，0のときはそうでないことを表す。

問1 ア、イ


```
(01) 表示する("自然数を入力してください")
(02) n = 【自然数を外部からの入力】
(03) sosuhantei = 1
(04) i = 2
(05) num = n
(06) i < num の間繰り返す:
(07)     もし n % i == [ア] ならば:
(08)         sosuhantei = 0
(09)         i = [イ]
(10)     i = i + 1
(11) もし n == 1 ならば:
(12)     表示する(n, "は素数ではありません")
(13) そうでなくもし sosuhantei == 1 ならば:
(14)     表示する(n, "は素数です")
(15) そうでなければ:
(16)     表示する(n, "は素数ではありません")
```

図1　自然数 n が素数かどうかを判定するプログラム

[イ] の解答群

⓪ 0
① 1
② num % 2
③ num

●例でnに4を入れる

●num=nなので4が入る

●i<numなので2<4から始まり
4<4になるまで繰り返す

●見ていく中でn%i==0ならば
sosuhanteiが0になる

●イは無理やりループを終わらせる命令
4<4にしたいのでiの値を4にしたい

答え　ア　0　イ③

問2

T：それでは次に 2 から n の間にある素数を探索して，その素数を小さい順に表示するプログラムを作成してみましょう。

S：わかりました。n は大きくても 1000 までを想定して，図 2 のようなプログラムを作成しました。なお配列 Prime[1000] の 1000 個の要素はすべて 0 で初期化し，素数と判定された自然数を順に格納していくようにしました。

```
(01) 表示する("2 以上 1000 以下の自然数を入力してください")
(02) n = 【2 以上 1000 以下の自然数を外部からの入力】
(03) Prime[1000] = [0, 0, 0, …, 0]
(04) i = 2
(04) j = 0
(05) num = 2
(06) num <= n の間繰り返す:
(07)     sosuhantei = 1
(08)     i < num の間繰り返す:
(09)         もし num % i == [ア] ならば:
(10)             sosuhantei = 0
(11)             i = [イ]
(12)         i = i + 1
(13)     もし sosuhantei == 1 ならば:
(14)         Prime[j] = [ウ]
(15)         j = j + 1
(16)     num = num + 1
(17) 表示する("1 から ", n, "までの素数:")
(18) i = 0
(19) Prime[i] [エ] [オ] の間繰り返す:
(20)     表示する(Prime[i])
(21)     i = i + 1
```

図 2　2 から n までの素数を小さい順に表示するプログラム

●primeに素数判定された数字を入れていく

●nに3といれる＝2から3までの素数がprimeに入る

●num<=nなので2<=3から始めり4<=3の間繰り返す

●2<=3から始まる

問2 ウ　エ　オ

T：それでは次に2からnの間にある素数を探索して，その素数を小さい順に表示するプログラムを作成してみましょう。

S：わかりました。nは大きくても1000までを想定して，図2のようなプログラムを作成しました。なお配列 Prime[1000] の1000個の要素はすべて0で初期化し，素数と判定された自然数を順に格納していくようにしました。

```
(01) 表示する("2以上1000以下の自然数を入力してください")
(02) n = [2以上1000以下の自然数を外部からの入力]
(03) Prime[1000] = [0, 0, 0, …, 0]
(04) i = 2
(04) j = 0
(05) num = 2
(06) num <= n の間繰り返す:
(07)     sosuhantei = 1
(08)     i < num の間繰り返す:
(09)         もし num % i == [ア] ならば:
(10)             sosuhantei = 0
(11)             i = [イ]
(12)     i = i + 1
(13)     もし sosuhantei == 1 ならば:
(14)         Prime[j] = [ウ]
(15)         j = j + 1
(16)     num = num + 1
(17) 表示する("1 から ", n, "までの素数:")
(18) i = 0
(19) Prime[i] [エ] [オ] の間繰り返す:
(20)     表示する(Prime[i])
(21)     i = i + 1
```

図2　2からnまでの素数を小さい順に表示するプログラム

行番号	n	num	i	j
(06)	3	2	2	0
(08)	i<numの間繰り返す(×) 2<2（×）→13行目へ			
(14)	Prime(j)=prime(0)=(　　)			
(15)				1
(16)		3		
(06)	num<=n（3<=3)(○)			
(08)	2<3の間繰り返す(○) num			
(13)	Prime(j)=prime(1)=(　　)			

●今primeには[2,3,0,0,0・・・0]が入っている

これを順番に表示するには

0になるまで繰り返したらいいので
Prime[i]!=0が入る
（0と等しくない間は繰り返し表示）

問3　素数判定の考え方

問3　次の会話文を読み，［カ］・［ケ］・［コ］・［サ］に当てはまる数字をマークせよ。また，空欄［キ］・［ク］に入れるのに最も適当なものを，後の解答群のうちから一つずつ選べ。

T：それでは今までのプログラムを改良して，なるべく割り算する回数が少なくなるように工夫しましょう。

S：まず偶数の素数は［カ］だけなので，［カ］以外の偶数は調べる必要はありません。あとはすべての自然数で割るのではなく，Prime[1000]にそれまで格納された素数で割っていけば十分でしょうか。

T：その通り。あと一息です。実はある自然数が素数でないとき，その自然数の平方根以下である2以上の正の約数が必ず存在するので，Prime[1000]にそれまでに格納された「すべての素数」で割らなくても十分です。それも含め最適化したプログラムが図3になります。なお2は配列Prime[1000]にすでに格納しておき，変数counterは割り算をした回数を調べるために追加した変数です。

●例えば36を例にとる

●2で割り切れるのは2×18

●3で割り切れるのは3×12

●4で割り切れるのは4×9

●5で割れるものはない

●6で割り切れるのは6×6

ここまで見たら9や12は
見なくてもいい

6×6<=36までみれば十分という考え
素数の中だけでいくと5×5<=36まで

問3 キ

問3　次の会話文を読み，［カ］・［ケ］・［コ］・［サ］に当てはまる数字をマークせよ。また，空欄［キ］・［ク］に入れるのに最も適当なものを，後の解答群のうちから一つずつ選べ。

T：それでは今までのプログラムを改良して，なるべく割り算する回数が少なくなるように工夫しましょう。

S：まず偶数の素数は［カ］だけなので，［カ］以外の偶数は調べる必要はありません。あとはすべての自然数で割るのではなく，Prime[1000]にそれまで格納された素数で割っていけば十分でしょうか。

T：その通り。あと一息です。実はある自然数が素数でないとき，その自然数の平方根以下である2以上の正の約数が必ず存在するので，Prime[1000]にそれまでに格納された「すべての素数」で割らなくても十分です。それも含め最適化したプログラムが図3になります。なお2は配列Prime[1000]にすでに格納しておき，変数counterは割り算をした回数を調べるために追加した変数です。

●もう一つの考えとしてはルートをつける
√36以下の数字まで

√36は6になるので6の倍数のまでみたらいいという考え

6×6<=36までみれば十分という考え
素数の中だけでいくと5×5<=36まで

答え　キ　prime[i]＊prime[i]＜＝num

問3 カ

問3　次の会話文を読み，カ・ケ・コ・サに当てはまる数字をマークせよ。また，空欄キ・クに入れるのに最も適当なものを，後の解答群のうちから一つずつ選べ。

T：それでは今までのプログラムを改良して，なるべく割り算する回数が少なくなるように工夫しましょう。

S：まず偶数の素数はカだけなので，カ以外の偶数は調べる必要はありません。あとはすべての自然数で割るのではなく，Prime[1000]にそれまで格納された素数で割っていけば十分でしょうか。

T：その通り。あと一息です。実はある自然数が素数でないとき，その自然数の平方根以下である2以上の正の約数が必ず存在するので，Prime[1000]にそれまでに格納された「すべての素数」で割らなくても十分です。それも含め最適化したプログラムが図3になります。なお2は配列Prime[1000]にすでに格納しておき，変数counterは割り算をした回数を調べるために追加した変数です。

●素数の中で偶数は2のみ

答え　カ　2

問3 ケ


```
(01) 表示する("2 以上の自然数を入力してください")
(02) n = 【2 以上の自然数を外部からの入力】
(03) Prime[1000] = [2, 0, 0, 0, …, 0]
(04) j = 1
(05) counter = 0
(06) num = 3
(07) num <= n をみたす間繰り返す:
(08)   sosuhantei = 1
(09)   i = 1
(10)   もし i < j かつ sosuhantei == 1 かつ [キ] ならばその間
       繰り返す:
(11)     counter = counter + 1
(12)     もし num % [ク] == 0 ならば:
(13)       sosuhantei = 0
(14)     i = i + 1
(15)   もし sosuhantei == 1 ならば:
(16)     Prime[j] = [ウ]
(17)     j = j + 1
(18)   num = num + [ケ]
(19) 表示する("1 から", n, "までの素数:")
(20) i = 0
(21) Prime[i] [エ] [オ] の間繰り返す:
(22)   表示する(Prime[i])
(23)   i = i + 1
(24) 表示する("割り算を行った回数:", counter)
```

図3　なるべく割り算をする回数を少なくした素数判定・表示プログラム

●n=9を例にする

行番号	n	num	i	j
(06)まで	9	3	1	1
(07)	num<=9を満たす間 3<=9(○)			
(10)	i<j　1<1（×）→15行目へ			
(16)	Prime(j)=(num)→prime[2,3,0,0・・]			
(17)				2
(18)	num=num+(　　)			
(18)	2　5			
(07)	num<=n　5<=9(○)			
(10)	i<j(○)　prime[i]×prime[i]<=num 3　×　3　<=5（×）→15行目へ			

3×3＜＝9に初めてあてはまるのはnumが9のときときことのとき初めてcoountに+1される